The Association for Computers and the Humanities (ACH)

The Association for Computational Linguistics (ACL)

The Association for Literary and Linguistic Computing (ALLC)

Guidelines for Electronic Text Encoding and Interchange

Edited by C. M. Sperberg-McQueen and Lou Burnard

TEI P3 Text Encoding Initiative Chicago, Oxford

16 May 1994

Contents

In memoriam
Donald E. Walker
22 November 1928 - 26 November 1993

Note

These Guidelines are the result of over five years' effort by members of the research and academic community within the framework of an international cooperative project called the Text Encoding Initiative (TEI), established in 1987 under the joint sponsorship of the Association for Computers and the Humanities, the Association for Computational Linguistics, and the Association for Literary and Linguistic Computing.

The impetus for the project came from the humanities computing community, which sought a common encoding scheme for complex textual structures in order to reduce the diversity of existing encoding practices, simplify processing by machine, and encourage the sharing of electronic texts. It soon became apparent that a sufficiently flexible scheme could provide solutions for text encoding problems generally. The scope of the TEI was therefore broadened to meet the varied encoding requirements of any discipline or application. Thus, the TEI became the only systematized attempt to develop a fully general text encoding model and set of encoding conventions based upon it, suitable for processing and analysis of any type of text, in any language, and intended to serve the increasing range of existing (and potential) applications and use.

What is published here is a major milestone in this effort. It provides a single, coherent framework for all kinds of text encoding which is hardware-, software- and application-independent. Within this framework, it specifies encoding conventions for a number of key text types and features. The ongoing work of the TEI is to extend the scheme presented here to cover additional text types and features, as well as to continue to refine its encoding recommendations on the basis of extensive experience with their actual application and use.

We therefore offer these Guidelines to the user community for use in the same spirit of active collaboration and cooperation with which they have so far been developed. The TEI is committed to actively supporting the wide-spread and large-scale use of the Guidelines which, with the publication of this volume, is now for the first time possible. In addition, we anticipate that users of the TEI Guidelines will in some instances adapt and extend them as necessary to suit particular needs; we invite such users to engage in the further development of the Guidelines by working with us as they do so.

Like any standard which is actually used, these Guidelines do not represent a static finished work, but rather one which will evolve over time with the active involvement of its community of users. We invite and encourage the participation of the the user community in this process, in order to ensure that the TEI Guidelines become and remain useful in all sorts of work with machine-readable texts.

This document was made possible in part by financial support from the U.S. National Endowment for the Humanities, an independent federal agency; Directorate General XIII of the Commission of the European Communities; the Andrew W. Mellon Foundation; and the Social Science and Humanities Research Council of Canada. Direct and indirect support has also been received from the University of Illinois at Chicago, the Oxford University Computing Services, the University of Arizona, the University of Oslo, Queen's University (Kingston, Ont.), Bellcore (Bell Communications Research), the Istituto di Linguistica Computazionale (C.N.R.) (Pisa), EDR (the Japan Electronic Dictionary Research Institute, Tokyo), the British Academy, and Ohio State University, as well as the employers and host institutions of the members of the TEI working committees and work groups listed in the acknowledgements.

The production of this document has been greatly facilitated by the willingness of many software vendors to provide us with evaluation versions of their products. Most parts of this text

have been processed at some time by almost every currently available SGML-aware software system. In particular, we gratefully acknowledge the assistance of the following vendors:

- Berger-Levrault AIS s.a. (for Balise).
- E2S n.v. (for E2S Advanced SGML Editor)
- Electronic Book Technology (for DynaText),
- SEMA Group and Yard Software (for Mark-It and Write-It),
- Software Exoterica (for CheckMark and Xtran),
- SoftQuad, Inc., (for Author/Editor and RulesBuilder),
- Xerox Corporation (for Ventura Publisher)

Details of the software actually used to produce the current document are given in the colophon at the end of the work.

Acknowledgments

Many people have given of their time, energy, expertise, and support in the creation of this document; it is unfortunately not possible to thank them all adequately. Below are listed those who have served as formal members of the TEI's Work Groups and Working Committees during its six-year history; others not so officially enfranchised also contributed much to the quality of the result.

The editors take this opportunity to acknowledge our debt to those who have patiently endured and corrected our misunderstandings of their work; we hope that they will feel the wait has not been in vain. For any errors and inconsistencies remaining, we must accept responsibility; any virtue in what is here presented, we gladly ascribe to the energies of the keen intellects listed below.

C. M. Sperberg McQueen and Lou Burnard

TEI Working Committees (1990-1993)

Note: Not all members listed were able to serve throughout the development of the Guidelines.

1. Committee on Text Documentation:
   Chair: Dominik Wujastyk (Wellcome Institute for the History of Medicine)
   Members 1990-1992: J. D. Byrum (Library of Congress); Marianne Gaunt (Rutgers University); Richard Giordano (Manchester University); Barbara Ann Kipfer (Independent Consultant); Hans Jørgen Marker (Danish Data Archive, Odense); Marcia Taylor (University of Essex);
2. Committee on Text Representation
   Chair: Stig Johansson (University of Oslo)
   Members 1990-1992: Roberto Cencioni (Commission of the European Communities); David R. Chesnutt (University of South Carolina); Robin C. Cover (Dallas Theological Seminary); Steven J. DeRose (Electronic Book Technology Inc); David G. Durand (Boston University); Susan M. Hockey (Oxford University Computing Service); Claus Huitfeldt (University of Bergen); Francisco Marcos-Marin (University Madrid); Elli Mylonas (Harvard University); Wilhelm Ott (University of Tübingen); Allen H. Renear (Brown University); Manfred Thaller (Max-Planck-Institut für Geschichte, Göttingen)
3. Committee on Text Analysis and Interpretation
   Chair: D. Terence Langendoen (University of Arizona)
   Members 1990-1992: Robert Amsler (Bell Communications Research); Stephen Anderson (Johns Hopkins University); Branimir Boguraev (IBM T. J. Watson Research Center); Nicoletta Calzolari (University of Pisa); Robert Ingria (Bolt Beranek Newman Inc); Winfried Lenders (University of Bonn); Mitch Marcus (University of Pennsylvania); Nelleke Oostdijk (University of Nijmegen); William Poser (Stanford University); Beatrice Santorini (University of Pennsylvania); Gary Simons (Summer Institute of Linguistics); Antonio Zampolli, University of Pisa.
4. Committee on Metalanguage and Syntax
   Chair: David T. Barnard (Queen's University); David G. Durand (Boston University); Jean-Pierre Gaspart (Associated Consultants and Software Engineers sa/nv); Nancy M. Ide (Vassar College); Lynne A. Price (Software Exoterica / Xerox PARC); Frank Tompa (University of Waterloo); Giovanni Battista Varile (Commission of the European Communities).

In addition, the two TEI editors served ex officio on each committee.

Following publication of the first draft of the TEI Guidelines (P1) in November 1990, a number of specialist work groups were charged with responsibility for drafting revisions and extensions, which, together with material already presented in P1, constitute the basis of the present work.

In addition, many members of the work groups listed below met on three occasions to review the emerging proposals in detail as members of the TEI Technical Review Committee. These meetings, held in Myrdal Norway (December 1991), Chicago (June 1992) and Oxford (March 1993), were largely responsible for the technical content and organization of the present work. Attendants at these meetings are starred in the list below.

TR1 Character sets Chair: Harry Gaylord* (University of Groningen); Syun Tutiya* (Chiba University).

TR2 Text Criticism Chair: Peter Robinson* (Oxford University); David Chesnutt* (University of South Carolina); Robin Cover* (Dallas Theological Seminary); Robert Kraft (University of Pennsylvania); Peter Shillingsburg (Mississippi State University).

TR3 Hypertext and hypermedia Chair: Steven J. DeRose* (Electronic Book Technologies Inc); David Durand (Boston University); Edward A. Fox (Virginia State University); Eve Wilson (University of Kent).

TR4 Formulæ, Tables, Figures, and Graphics Chair: Paul Ellison* (University of Exeter); Anders Berglund (Independent Consultant); Dale Waldt (Thompson Professional Publishing).

TR6 Language corpora Chair: Douglas Biber* (University of Northern Arizona); Jeremy Clear (Birmingham University); Gunnel Engwall (University of Stockholm).

TR9 Manuscripts and Codicology Chair: Claus Huitfeldt* (University of Bergen); Dino Buzzetti (University of Bologna); Jacqueline Hamesse (University of Louvain); Mary Keeler (Georgetown University); Christian Kloesel (Indiana University); Allen Renear* (Brown University); Donald Spaeth (Glasgow University).

TR10 Verse Chair: David Robey* (University of Manchester); Elaine Brennan* (Brown University); David Chisholm (University of Arizona); Willard McCarty (University of Toronto).

TR11 Drama and performance texts Chair: Elli Mylonas* (Harvard University); John Lavagnino* (Brandeis University); Rosanne Potter (University of Iowa).

TR12 Literary prose Chair Thomas N. Corns* (University of Wales); Christian Delcourt (University of Liège;).

AI1 Linguistic Description Chair: D. Terence Langendoen* (University of Arizona); Stephen R. Anderson (Johns Hopkins University); Nicoletta Calzolari (University of Pisa); Geoffrey Sampson* (University of Sussex); Gary Simons* (Summer Institute of Linguistics).

AI2 Spoken text Chair: Stig Johansson* (University of Oslo); Jane Edwards (University of California at Berkeley); Andrew Rosta (University College London).

AI3 Literary studies Chair: Paul Fortier* (University of Manitoba); Christian Delcourt (University of Liège;); Ian Lancashire (University of Toronto); Rosanne Potter (University of Iowa); David Robey* (University of Manchester).

AI4 Historical studies Chair: Daniel Greenstein* (University of Glasgow); Peter Denley (Queen Mary Westfield College, London); Ingo Kropac (University of Graz); Hans Jørgen Marker (Danish Data Archive, Odense); Jan Oldervoll (University of Tromsø); Kevin Schurer (University of Cambridge); Donald Spaeth (Glasgow University); Manfred Thaller (Max-Planck-Institut für Geschichte, Göttingen). [1]

AI5 Print dictionaries Chairs: Robert Amsler* (Bell Communications Research) and Nicoletta Calzolari (University of Pisa); Susan Armstrong-Warwick (University of Geneva); John Fought (University of Pennsylvania); Louise Guthrie (University of New Mexico); Nancy M. Ide* (Vassar College); Frank Tompa (University of Waterloo); Carol Van Ess-Dykema (Department of Defense); Jean Veronis (University of Aix-en-Provence).

AI6 Machine Lexica Chair: Robert Ingria* (Bolt Beranek Newman Inc); Susan Armstrong-Warwick (University of Geneva); Nicoletta Calzolari (University of Pisa).

AI7 Terminological data Chair: Alan Melby* (Brigham Young University) Gerhard Budin (University of Vienna); Gregory Shreve (Kent State University); Richard Strehlow (Oak Ridge

[1] This Workgroup was jointly sponsored by the Association for History and Computing.

National Laboratory); Sue Ellen Wright (Kent State University).

Advisory Board

Members of the TEI Advisory Board during the life time of the project are listed below, grouped under the name of the organization represented.

American Anthropological Association: Chad McDaniel (University of Maryland).
American Historical Association: Elizabeth A. R. Brown (Brooklyn College, CUNY).
American Philological Association: Jocelyn Penny Small (Rutgers University).
American Philosophical Association: Allen Renear (Brown University).
American Society for Information Science: Clifford A. Lynch (University of California).
Association for Computing Machinery, Special Interest Group for Information Retrieval: 1989-93: Scott Deerwester (University of Chicago); 1993- : Martha Evens (Illinois Institute of Technology).
Association for Documentary Editing: David Chesnutt (University of South Carolina).
Association for History and Computing: 1989-91: Manfred Thaller, Max-Planck-Institut fü Geschichte, Göttingen; 1991- : Daniel Greenstein (Glasgow University).
Association Internationale Bible et Informatique 1989-93: Wilhelm Ott (University of Tübingen); 1993- : Winfried Bader (University of Tübingen).
Canadian Linguistic Association: Anne-Maria di Sciullo (Université du Québec à Montréal).
Dictionary Society of North America: Barbara Ann Kipfer (independent consultant).
AAP Electronic Publishing Special Interest Group: 1989-92: Betsy Kiser (OCLC); 1992- : Deborah Bendig and Andrea Keyhani (OCLC).
International Federation of Library Associations and Institutions: J. D. Byrum Jr. (The Library of Congress).
Linguistic Society of America: Stephen Anderson (The Johns Hopkins University).
Modern Language Association: Randall Jones (Brigham Young University) and Ian Lancashire (University of Toronto).

Steering Committee Membership

Members of the Steering Committee of the TEI during the preparation of this work were:

Association for Computational Linguistics:

- 1987-1993: Robert A. Amsler (Bell Communications Research);
- 1987-1993: Donald E. Walker (Bell Communications Research);
- 1993- : Susan Armstrong-Warwick (University of Geneva);
- 1994- : Judith Klavans (Columbia University).

Association for Computers and the Humanities:

- 1987- : Nancy M. Ide (Vassar College);
- 1987-1994: C. M. Sperberg-McQueen (University of Illinois at Chicago);
- 1994- : David Barnard (Queen's University).

Association for Literary and Linguistic Computing:

- 1987- : Susan M. Hockey (Center for Electronic Texts in the Humanities);
- 1987- : Antonio Zampolli (University of Pisa).

Changes from TEI P1 to TEI P3

This list gives a partial indication of the major changes from Versions 1 and 2 of these Guidelines (issued by the TEI as drafts under the document numbers TEI P1 and TEI P2, the latter released in chapters between March 1992 and the end of 1993) to the current text.

Chapter 1 ('About These Guidelines') on p. 5: this chapter corresponds to chapter 1 of TEI P1 and TEI P2; it has been reorganized, revised, and expanded, and a new section explaining the notational conventions of this document has been added.

Chapter 2 ('A Gentle Introduction to SGML ') on p. 15: this is a slightly revised version of chapter 2 of TEI P1 and P2. Brief discussions of parameter entities and marked sections have been added, but no other changes of substance have been made.

Chapter 3 ('Structure of the TEI Document Type Definition') on p. 35: this chapter was introduced in TEI P2; the lists of classes and important parameter entities have been updated, and some declarations have been reordered; no other changes have been made.

Chapter 4 ('Characters and Character Sets') on p. 71: this chapter corresponds to some material in chapter 3 of TEI P1, but presents it in what is hoped to be a more accessible form. No substantive changes have been made since its publication as part of TEI P2.

Chapter 5 ('The TEI Header') on p. 77: this is a revised and much expanded version of chapter 4 of TEI P1. The overall structure of the TEI header has been retained, but most of the elements have been renamed to match a new set of naming conventions. The <encoding.declarations> element of TEI P1 has been split into the <encodingDesc> and <profileDesc> elements, the former concentrating on the process by which the electronic text has been encoded, the latter on the non-bibliographic characteristics of the text itself. A number of specialized declarations have been added to both these sections of the header, in order to allow the formal specification of important information about the text and its encoding.

Chapter 6 ('Elements Available in All TEI Documents') on p. 119: this chapter corresponds to sections 5.3 to 5.6, portions of 5.7, and 5.8 of the first public draft of these Guidelines (TEI P1). Changes made to this material in this version include:

- The individual sections have been reordered and reorganized.
- Highlighting and quotation marks are treated together.
- The tags for names and dates have been revised, and a separate additional tag set has been provided for detailed analysis of names and dates (chapter 20 ('Names and Dates') on p. 487).
- The tags for simple editorial interventions have been revised; the new set includes several complementary pairs of elements, so that the encoder is consistently given the choice of recording the original text, or an editorial modification of it, as data content, and the other as an optional attribute value.
- The tags for bibliographic references have been renamed (from <citn> to <bibl>, etc.) and a new form (<biblFull>), corresponding to the structure of the TEI header, has been added.
- The treatment of canonical reference systems has been thoroughly revised and the discussion is now supplemented by discussions in chapter 5 ('The TEI Header') on p. 77, and chapter 32 ('Algorithm for Recognizing Canonical References') on p. 641.

Chapter 7 ('Default Text Structure') on p. 183: this chapter corresponds to section 5.2 of TEI P1. Changes made to this material in this version include:

- The theoretical discussion of alternative methods of constructing a tag set for overall text structure has been suppressed.
- The tags for elements of a title-page have been renamed.

- The specialized tags for divisions of front matter and back matter (<foreword>, <acknowledgements>, <dedication>, <colophon>, etc.) have been deleted; like those of the text body, these elements may be tagged with generic elements.
- In addition to numbered elements, the current draft also allows for un-numbered generic s.
- The treatment of collections and anthologies is explicitly discussed, building on section 7.2 of TEI P1, and the <group> element is introduced to deal with them.

Chapter 8 ('Base Tag Set for Prose') on p. 211, chapter 9 ('Base Tag Set for Verse ') on p. 213, and chapter 10 ('Base Tag Set for Drama') on p. 227: these correspond to the subparts of section 7.3 of TEI P1, but have been completely redesigned and rewritten from scratch.

Chapter 11 ('Transcriptions of Speech') on p. 249: this chapter first appeared in TEI P2; it has been revised here to match changes in the overall design of the Guidelines since its publication. Most importantly, this tag set now uses the default text-structure elements described in chapter 7 ('Default Text Structure') on p. 183, and the methods for handling overlap and other time-specific information have been revised to make use of the techniques described in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331.

Chapter 12 ('Print Dictionaries') on p. 269: the tag set presented in this chapter is a complete revision of that described in section 7.4 of TEI P1, and the chapter itself was entirely rewritten from scratch.

Chapter 13 ('Terminological Databases') on p. 311: this chapter was first published in December, 1993, as part of TEI P2. Since that publication, it has been revised slightly for the sake of consistency with the rest of the Guidelines and with the work of Technical Committee 37 of the International Organization for Standardization (ISO) on ISO DIS 12 200.

Chapter 14 ('Linking, Segmentation, and Alignment') on p. 331: this chapter corresponds to sections 5.7 ("Links and Cross References") and sections 6.2.3 through 6.2.5 ("Alignment of Multiple Analyses," etc.) of TEI P1. Changes made to this material in this version include:

- The <xref> element of P1 has been split into the two elements <ptr> and <xptr>, of which the former is used to point at IDs within the document and the latter for pointing outside the document or for pointing at passages without IDs in the current document.
- The elements <ref> and <xref> have been added to provide pointer elements which can accept character content, for cases in which the pointing phrase of the source text cannot be reconstructed algorithmically.
- The "extended pointer syntax" has been substantially revised and systematized; the syntax and semantics of extended pointers have been defined more precisely.
- The <unit> and <level> elements defined by P1 for implicit alignment of multiple levels of analysis have been dropped; in their stead, the revised feature structure elements should be used. These are defined in chapter 16 ('Feature Structures') on p. 397.
- The elements <alignment>, <al.map>, <al.ptr>, <al.list>, <al.range>, defined in P1 for explicit alignment of multiple texts or analyses, have been replaced by the <link>, <linkGrp>, <corresp> and <correspGrp> elements. Because the <link> and <corresp> elements can point at multiple targets, there is no need for special alignment pointers, alignment lists, or alignment range elements.
- The elements <link> and <correspond> may be used in connection with <xptr> to align elements in external entities or passages which do not bear SGML identifiers.

Since the publication of this chapter in TEI P2, it has been revised, the section on alternation has been added, new examples have been introduced, and the extended pointer syntax has been revised. The extended pointer syntax is now also used to specify canonical reference systems, as well as in the <xptr> and <xref> elements.

Chapter 15 ('Simple Analytic Mechanisms') on p. 381: the bulk of this chapter is new, though some parts of its substance derive from chapter 6 of TEI P1. The global ana and inst attributes have been added, to simplify the notation for simple forms of alignment between text and analyses; the elements and <interp> have been introduced, to simplify the specification of analyses which do not require the structural rigor of feature structures.

Chapter 16 ('Feature Structures') on p. 397: this chapter derives from sections 6.2.1 and 6.3 of TEI P1. In its broad outlines, the feature structure notation introduced there is retained. The

most important changes include these:

- Some elements have been renamed.
- Feature names and feature structure names are now represented as attribute values, rather than as embedded subelements.
- The treatment of Boolean logic has been substantially changed.

Chapter 17 ('Certainty and Responsibility') on p. 435: this chapter was introduced in TEI P2; its wording has been revised slightly since then, but the tags described remain the same.

Chapter 18 ('Transcription of Primary Sources') on p. 443: this chapter presents new material on the use of the core tags for editorial intervention, and on specialized problems in the transcription of primary source material, especially manuscripts.

Chapter 19 ('Critical Apparatus') on p. 467: this chapter is a substantial revision of section 5.10 ("Critical Apparatus") of TEI P1. The major changes include the following:

- The single end-point attachment method of encoding critical apparatus has been dropped.
- A new method of apparatus encoding, the location-referenced method, has been introduced to simplify the transcription of existing critical editions.
- The problem of subvariation is treated more explicitly.
- The <witList> element has been introduced for the purpose of identifying all the witnesses whose readings are recorded in the apparatus.
- The treatment of detailed information about a particular reading in a particular witness (in the <witDetail> element) has been changed somewhat.

Chapter 20 ('Names and Dates') on p. 487: this chapter is new in this version.

Chapter 22 ('Tables, Formulae, and Graphics') on p. 523: this chapter replaces section 5.9 of TEI P1; it provides small but usable tag set for tables, and describes in much more detail the process of including graphical information (figures, illustrations, etc.) in TEI-encoded texts.

Chapter 23 ('Language Corpora') on p. 537: this chapter builds on section 7.2 of TEI P1, but its contents are largely new. The tag set described here provides much fuller methods for documenting text type, subject area, and demographic characteristics of speakers, listeners, authors, etc. associated with the texts of a corpus.

Chapter 24 ('The Independent Header') on p. 559: this chapter was introduced in TEI P2; it has been slightly revised since.

Chapter 25 ('Writing System Declaration') on p. 571: this chapter derives from the writing system declaration described in chapter 3 ("Characters and Character Sets") of TEI P1. The structure of the WSD has been changed slightly, and the chapter now gives an explicit account of the semantics of specifying base character sets, entity sets, or WSDs, and of modifying them using the <exceptions> element.

Chapter 26 ('Feature System Declaration') on p. 589: this chapter is new in this version of these Guidelines.

Chapter 27 ('Tag Set Documentation') on p. 601: this chapter first appeared in TEI P2; it has not changed substantially since.

Chapter 28 ('Conformance') on p. 611, chapter 29 ('Modifying the TEI DTD') on p. 619, chapter 30 ('Rules for Interchange ') on p. 627, chapter 31 ('Multiple Hierarchies') on p. 633, and chapter 32 ('Algorithm for Recognizing Canonical References') on p. 641: these chapters are all new in the current version of these Guidelines (though the mechanisms of modifying the TEI DTDs described in chapter 29 ('Modifying the TEI DTD') on p. 619 remain the same as those described in chapter 8 of TEI P1). The definition of conformance provided in this version of these Guidelines differs from that of TEI P1 primarily in making more explicit the nature of the requirement that extensions to the tag set be documented, in specifying the nature of the DTD modifications allowed in TEI-conformant documents, and in completely divorcing the issue of TEI-conformance from that of the character sets used in the document.

The alphabetical reference list of classes, entities, and elements was introduced in TEI P2; in this version, slightly fuller information is given. For element classes, lists of members are given which include members of all subclasses, and the declarations of the a-dot and m-dot parameter entities for the class are reproduced. The files in which entities and elements are declared are also given.

Chapter 39 (‘Formal Grammar for the TEI-Interchange-Format Subset of SGML’) on p. 993: this chapter appeared in TEI P2 and has not been revised for this version of these Guidelines.

Part I

Introduction

Chapter 1

About These Guidelines

These Guidelines have been developed by the Text Encoding Initiative (TEI); see 1.3 ('Historical Background') on p. 12. They are addressed to anyone who works with any text in electronic form. They provide means of representing those features of a text which need to be identified explicitly in order to facilitate processing of the text by computer programs. In particular, they specify a set of markers (or *tags*) which may be inserted in the electronic representation of the text, in order to mark the text structure and other textual features of interest. Without such explicit markers, many important features remain difficult to locate by mechanical means such as computer programs, and thus difficult to process effectively. The process of inserting such explicit markers for implicit textual features is often called "markup" or "tagging", and the term *encoding scheme* or *markup language* denotes the rules which govern the use of markup in a set of encodings.

The Guidelines formulated in this document are intended for use in interchange between individuals and research groups using different programs and computer systems over a broad range of applications. Since they contain an inventory of the features most often found useful for text processing, the Guidelines also provide help to those creating texts in electronic form. They can also be used for the local storage of text which is to be processed with multiple software packages requiring different input formats. The Guidelines apply to texts in any natural language, of any date, in any literary genre or text type, without restriction on form or content. They treat both continuous materials ("running text") and discontinuous materials such as dictionaries and linguistic corpora. Though principally directed to the needs of the scholarly research community, the Guidelines are not restricted to esoteric academic applications. They should also be useful for librarians who maintain and document electronic materials, as well as for publishers and others creating or distributing electronic texts. Although they focus on problems of representing in electronic form texts which already exist in traditional media, these Guidelines should also be useful for the creation of electronic texts. They are adequate to, but not limited by, existing practices.

The rules and recommendations made in the these Guidelines conform to ISO 8879, which defines the Standard Generalized Markup Language (SGML), and make reference to ISO 646, which defines a standard seven-bit character set in terms of which the recommendations on character-level interchange are formulated. For more information on SGML see chapter 2 ('A Gentle Introduction to SGML ') on p. 15. This document provides the authoritative statement of the requirements and usage of the TEI encoding scheme. Although it includes numerous small examples, it must be stressed that it is intended as a reference manual and that readers unfamiliar with SGML or text markup in general will find it difficult to learn the encoding scheme by reading this document alone. This document will be complemented by a series of tutorials in text encoding (document TEI U1 et seq.) and a case book of extended examples with discussion of the rationale for various markup choices (TEI T1). [1] Readers seeking an introduction to text markup and the use of the TEI encoding scheme in a specific area should consult an appropriate tutorial;

[1] TEI documents bear identifying numbers which indicate the provenance of the document (here simply "TEI", in other cases the TEI work group number, e.g. "TEI AI5"), the type of document (here "U" and "T", meaning 'users' guide' or 'users' manual' and 'sample text(s)'), and a sequential number. The TEI document number of the document in hand is TEI P3 (for 'TEI public proposal number 3').

those already familiar with the scheme and interested in seeing examples of its application should consult the case book. The remainder of this chapter comprises three sections. The first gives an overview of the structure and notational conventions used throughout the document. The second enumerates the design principles underlying the TEI scheme and the application environments in which it may be found useful. Finally, the third section gives a brief account of the origins and development of the Text Encoding Initiative itself.

1.1 Structure and Notational Conventions of this Document

1.1.1 Structure

Part I provides some relevant background information about the Guidelines themselves (in this chapter); a brief technical review of SGML (chapter 2 ('A Gentle Introduction to SGML ') on p. 15); and a description of how the TEI document type definition (DTD) is organized (chapter 3 ('Structure of the TEI Document Type Definition') on p. 35).

Part II provides a systematic treatment of issues common to all text types: character representation (chapter 4 ('Characters and Character Sets') on p. 71); in-file documentation of the text (chapter 5 ('The TEI Header') on p. 77); tags for text features found in all sorts of text: lists, notes, emphasis, quotations, cross-references, technical terms, names, dates, numbers, etc. (chapter 6 ('Elements Available in All TEI Documents') on p. 119); and a definition for the default structure of all TEI documents (chapter 7 ('Default Text Structure') on p. 183). Part III documents various *base tag sets*: these include specialized tags for prose, for verse, for drama and other performance materials, for spoken materials, as well as for letters and memoranda, printed dictionaries, and terminological data. Additional sections discuss user-defined and mixed base tag sets. An instance of the TEI DTD must use one and only one base tag set, unless one of the "mixed" bases is used. Part IV documents various *additional tag sets*, which may be included or excluded, as appropriate. Topics covered include a variety of approaches to the analysis and interpretation of texts, and include representations for hypertextual links and other non-hierarchic structures, as well as specialized tags for the encoding of critical editions and language corpora.

Part V defines certain specialized *auxiliary document types*, used to encode information about the way that texts have been encoded, specifically: the TEI header regarded as a distinct document; the TEI Writing System Declaration; the Feature System declaration; and the Tag Set Documentation. Part VI contains a number of technical discussions of a more specialist interest. Topics covered include the notion of formal *conformance* to the TEI Guidelines; the controlled user-modification of the TEI DTD; practical aspects of the use of TEI markup both in local processing and in interchange; and the relationship of TEI markup to other markup standards. Part VII consists of an alphabetical reference list of all elements and element classes defined in the TEI encoding scheme. Part VIII provides further reference material: specifically, a description of how to obtain current versions of the full TEI DTDs and the set of standard Writing System Declarations, a sample Feature System Declaration for basic grammatical annotation, sample tag documentation, and a formal grammar for the subset of SGML used in the TEI interchange format. In the back matter, a bibliography lists works cited in the text of the Guidelines. A mechanically generated index is also provided, which can serve, it is hoped, as a finding aid for the use of the Guidelines.

1.1.2 Notational Conventions

This section describes the typographic and stylistic conventions used throughout this document. The use of many terms and concepts which have not yet been defined is unavoidable in this section. All such terms and concepts will be explained in later chapters of Part I.

When SGML elements are mentioned in the text, the mentions take the form <name>, where "name" is the *generic identifier* of the element. Sample tags mentioned in the text are

displayed in the form <name att=value att2='value two'>. References to SGML attributes take the form attname, where "attname" is the name of the attribute. Where the elements and attributes thus mentioned are part of the TEI encoding scheme, they are included in the index. These Guidelines distinguish encoding practices, and SGML elements, which are required, recommended, or optional. The phrases "must", "is required to", etc., mark practices and tags which are required for TEI conformance. The phrases "should", "it is recommended that", "it is preferable to ...", etc., are used in describing practices which are recommended but not required for TEI conformance. Modal verbs like "may", "might", etc., mark practices which are strictly optional. Qualifying phrases like "if desired", "where appropriate", or "under some circumstances" are used when some tag or practice described may be desirable or acceptable under some circumstances and not under others. In the reference section in Part VII, elements and their attributes are all classed as one of:

required unconditionally required in a TEI-conformant document
mandatory when applicable required under the appropriate conditions; may be omitted if not applicable
recommended recommended unless there are good reasons, in the given circumstances, against it
recommended when applicable recommended under some circumstances (which should be clear from context)
optional strictly optional

This reference section includes cross-references to the chapter or section of the main text within which each element is discussed. Most sections of the main text in which elements are defined begin with a descriptive list of the elements concerned in the following format:

<tag> short description of the element marked by <tag>. Where appropriate this is followed by a list of significant non-global attributes for the element as follows:
  attribute description of the attribute's meaning or usage, optionally followed by a list of suggested or legal values:
    value1 meaning of value1
    value2 meaning of value2

Not all attributes are always included in these lists; those which are shared with other elements in a class are usually listed separately, and those of relatively specialized interest are usually listed only in the reference section. The values of the attribute are introduced with one of the following formulaic phrases:

Legal values include: The attribute cannot take values other than those given. Other values will cause SGML parsing errors. (This is used relatively rarely in these Guidelines.)
Suggested values include: The values listed constitute a set which should suffice for most purposes, and they should be used where appropriate. Developers of TEI-aware software should ensure that their software can process these values appropriately. In some cases, however, it is conceivable that other values might be necessary, so the SGML declaration for the attribute does not restrict legal values to those given. TEI-aware software should have reasonable fallback processing for values not in the list.
Sample values include: The attribute can take any value; those listed are provided simply as examples of the kind of value possible.

Each list of elements is followed by some discussion of its semantics and usage, followed by one or more examples, taken wherever possible from real texts, and presented in the following format:

```
<p>This paragraph contains an <hi rend=it>italicized phrase</hi>
```

All the examples are (or should be) legal SGML, but, because they are fragmentary, may not be parseable by SGML parsers without the required context. They also frequently make liberal use of white space to exhibit the logical structure of the SGML coding more clearly. Although this does not affect the SGML conformance of the examples, some users will prefer not to follow it in practice, since not all processors will ignore the extra white space. Examples may:

- show full start- and end-tags for all elements
- use empty end-tags (of the form </>) to close the most recently opened element

- omit end-tags (never start-tags) where they may legally be omitted; where this is done, it is normally mentioned in the accompanying text

Attribute values are given indifferently in single quotes or double quotes; unquoted attribute values are sometimes used where SGML requires no quotation marks. It should be noted that the examples demonstrate a variety of tagging styles, mostly aimed at making the tagging legible while also showing fairly explicitly where all elements begin and end. No claim is made or implied as to the appropriateness of the style adopted here for other purposes; in particular, those using SGML for local processing may often prefer to use empty end-tags more frequently than is shown in the examples, or to omit end-tags.

After the examples and usage notes, each section typically concludes with a DTD fragment containing the formal declarations for the elements described. Each DTD fragment is given a heading, and may contain element and attribute list declarations, entity declarations, parameter entity references, comments, and references to DTD fragments in other sections. The DTD fragments of a single chapter almost invariably belong to the same DTD file, the structure of which is typically described (with references to the included fragments) in one of the first or last sections of the chapter. The DTD fragments are identical to the DTDs distributed with these Guidelines, with the following exceptions:

- In the text, the DTD fragments appear in an order dictated by organization of this document; the actual DTD files may re-order the material slightly. This is indicated in the text by references from one DTD fragment to another.
- The DTD fragments in the text show the generic identifiers of all elements using the standard English names assigned in this document; the actual DTD files use parameter entities for all generic identifiers, so that elements can be conveniently renamed, as described in chapter 29 (‘Modifying the TEI DTD’) on p. 619.
- The actual DTD files include conditional marked sections surrounding the element and attribute list declaration for each element, to ensure that elements can conveniently be suppressed or redefined, as described in chapter 29 (‘Modifying the TEI DTD’) on p. 619. The fragments in the text suppress the marked-section-open and marked-section-close markup.

What appears in the text, therefore, as:

```
<!ELEMENT blort - O (farble+)>
```

will appear thus in the actual DTD file:

```
<![ %blort [
<!ELEMENT %n.blort - O ((%n.farble)+)>
]]>
```

For further discussion, see chapter 3 (‘Structure of the TEI Document Type Definition’) on p. 35, or chapter 29 (‘Modifying the TEI DTD’) on p. 619.

1.2 Underlying Principles and Intended Use

1.2.1 Design Principles of the TEI Scheme

The planning conference held at Vassar College in November, 1987 (see section 1.3 (‘Historical Background’) on p. 12) agreed on a number of principles concerning the basic design goals of the Text Encoding Initiative. These principles are expounded in various documents of the TEI (notably TEI ED P1 and TEI ED P2) and the interested reader is directed to those documents for further discussion.

Because of its roots in the humanistic research community, the TEI scheme is driven by its original goal of serving the needs of research, and is therefore committed to providing a maximum of comprehensibility, flexibility, and extensibility. More specific design goals of the TEI have been that the Guidelines should:

- provide a standard format for data interchange
- provide guidance for encoding of texts in this format

- support the encoding of all kinds of features of all kinds of texts studied by researchers
- be application independent

This has led to a number of important design decisions, such as:

- the choice of SGML and ISO 646
- the provision of a large predefined tag set
- a distinction between required, recommended, and optional encoding practices
- encodings for different views of text
- alternative encodings for the same text features
- mechanisms for user-defined extensions to the scheme

These goals and principles are expounded in more detail below. The goals of creating a common interchange format which is application independent require the definition of a specific markup syntax as well as the definition of a large predefined tag set. The syntax of the recommendations made in this document conforms to the international standard ISO 8879, which defines the Standard Generalized Markup Language; reference is also made to ISO 646, which defines a standard seven-bit character set. Full SGML document type declarations are provided for the scheme described in these Guidelines. The goal of providing guidance for text encoding requires that recommendations be made as to what textual features should be recorded in various situations. This mandate is fulfilled by the explicit specification, in the reference section for each tag, that the tag is *required*, *mandatory when applicable* but otherwise omissible, *recommended* generally, *recommended when applicable* but not always applicable, or *optional*. However, the TEI Guidelines make (with relatively rare exceptions) no suggestions or restrictions as to the relative importance of textual features. The philosophy of the Guidelines is "if you want to encode this feature, do it this way" — but very few features are mandatory.

The Guidelines have been written largely with a focus on text capture (i.e. the representation in electronic form of an already existing copy text in another medium) rather than text creation (where no such copy text exists). Hence the frequent use of terms like "transcription", "original", "copy text", etc. However, the Guidelines should be equally applicable to text creation, and the two terms *text creation* and *text capture* are often used interchangeably. Concerning text capture the TEI Guidelines do not specify a particular approach to the problem of fidelity to the source text and recoverability of the original; such a choice is the responsibility of the text encoder. The current version of these Guidelines, however, provides a more fully elaborated set of tags for markup of rhetorical, linguistic, and simple typographic characteristics of the text than for detailed markup of page layout or for fine distinctions among type fonts or manuscript hands.

In these Guidelines, no hard and fast distinction is drawn between "objective" and "subjective" information or between "representation" and "interpretation". These distinctions, though widely made and often useful in narrow well defined contexts, are perhaps best interpreted as distinctions between issues on which there is a scholarly consensus and issues where no such consensus exists. Such consensus has been, and no doubt will be, subject to change. The TEI Guidelines do not make suggestions or restrictions as to which of these features should be encoded. The use of the terms *descriptive* and *interpretive* about different types of encoding in the Guidelines is not intended to support any particular view on these theoretical issues, but reflects a purely practical division of responsibility between the two committees called Committee on Text Representation and Committee on Text Interpretation and Analysis. In general, the accuracy and the reliability of the encoding and the appropriateness of the interpretation is for the individual user of the text to determine. The Guidelines provide a means of documenting the encoding in such a way that a user of the text can know the reasoning behind that encoding, and the general interpretive decisions on which it is based. It is strongly recommended that the TEI header be used to give an account of these aspects of the encoding. The TEI header is described in chapter 5 ('The TEI Header') on p. 77. In many situations more than one view of a text is needed. No absolute recommendation to embody one specific view of text can apply to all texts and all approaches to them. The syntax of SGML ensures that some encodings can be ignored for some purposes. To enable encoding multiple views, these Guidelines not only treat a variety of text features, but they sometimes provide several alternative encodings for what appear to be identical textual phenomena. These Guidelines therefore offer the possibility of encoding many different views of the text, simultaneously if necessary. However, the Guidelines are built

on the assumption that there is a common core of textual features shared by virtually all texts and virtually all serious work on texts. This core set of tags is defined in Chapter 6 ('Elements Available in All TEI Documents') on p. 119. Beyond this core, many different elements can be encoded. In brief, the TEI Guidelines define a general-purpose encoding scheme which makes it possible to encode different views of text, possibly intended for different applications, serving the majority of scholarly purposes of text studies in the humanities. However, no predefined encoding scheme can serve all research purposes. Therefore, the TEI also provides means of modifying and extending the encoding scheme defined by the Guidelines (see chapter 29 ('Modifying the TEI DTD') on p. 619).

1.2.2 Intended Use

We envisage three primary functions for these Guidelines:

- guidance for individual or local practice in text creation and data capture;
- support of data interchange;
- support of application-independent local processing.

These three functions are so thoroughly interwoven in practice that it is hardly possible to address any one without addressing the others. However, the distinction provides a useful framework for discussing the possible role of the Guidelines in work with electronic texts.

1.2.2.1 Use in Text Capture and Text Creation

The description of textual features found in the chapters which follow should provide a useful checklist from which scholars planning to create electronic texts should select the subset of features suitable for their project. Problems specific to text creation or text "capture" have not been considered explicitly in this document. For purposes of the TEI interchange format and for use of SGML, it does not matter how a text is created or captured: it can be typed by hand, scanned from a printed book or typescript, read from a typesetter's tape, or acquired from another researcher who may have used another markup scheme (or no explicit markup at all). We include here only some general points which are often raised about SGML and the process of data capture. SGML can appear distressingly verbose, particularly when (as in these Guidelines) the names of tags and attributes are chosen for clarity and not for brevity. Editor macros and keyboard shorthands can allow a typist to enter frequently used tags with single keystrokes. Special-purpose software may be purchased which scans word-processor or scanner data and inserts SGML tags. SGML-aware software can help with maintaining the hierarchical structure of the document, and display the document with visual formatting rather than raw tags. The techniques described in chapter 29 ('Modifying the TEI DTD') on p. 619 may be used to give shorter names to the tags being used most often. It should also be noted that the examples in this text are chosen to exhibit the markup as compactly as possible, and thus have denser markup than will be typical in many texts. The SGML standard provides ways of abbreviating, omitting, or otherwise *minimizing* the amount of markup which need be explicitly provided in a text. They are all forbidden in the TEI interchange format because their use complicates processing; this does not however preclude their use in local processing, where this is felt appropriate or desirable.

1.2.2.2 Use for Interchange

When the TEI Guidelines are used for interchange, it is expected that researchers using other encoding schemes in their work will translate outgoing data from such schemes into the scheme described by these Guidelines, and similarly translate incoming data from the scheme described here into those used internally. For such translations to be carried out without loss of information, the scheme proposed here must be as expressive (in a formal sense) as any encoding scheme now known to be in wide use for textual research. To ensure that this is the case, a set of extension techniques is provided (see chapter 29 ('Modifying the TEI DTD') on p. 619) which makes possible the addition of extra tags, the renaming of existing tags and certain kinds of redefinition. Although the intention is to minimize the need for recourse to such extensions, they may be used

to accommodate the encoding of new or unanticipated textual features. To translate between any pair of encoding schemes implies:

1. identifying the sets of textual features distinguished by the two schemes;
2. determining where the two sets of features correspond;
3. creating a suitable set of mappings.

For example, to translate from encoding scheme X into the TEI scheme:

1. Make a list of all the textual features distinguished in X.
2. Identify the corresponding feature in the TEI scheme. There are three possibilities for each feature:
   (a) the feature exists in both X and the TEI scheme;
   (b) X has a feature which is absent from the TEI scheme;
   (c) X has a feature which corresponds with more than one feature in the TEI scheme.

   The first case is unproblematic. The second requires an extension to the TEI scheme, as described in chapter 29 ('Modifying the TEI DTD') on p. 619. The third requires that a consistent choice be made. The algorithm used to make that choice should be documented in the TEI header.
3. Using the table of equivalences so generated, a simple translation can be carried out between X and the TEI.

The ease with which this translation can be carried out will of course depend on the clarity and explicitness with which scheme X represents the features it encodes.

Translating from the TEI into scheme X follows the same pattern, except that if a TEI feature has no equivalent in X, and X cannot be extended, information must be lost in translation. Similar procedures may be followed where the TEI scheme is to be used as an interlanguage for interchange among several different sites or applications, although the degree of TEI-conformance may vary. In the simplest case, where two sites or individuals exchanging texts know each other and know or can inquire what equipment the other is using, these Guidelines serve primarily as documentation for a file format, which can be referred to without actually being transmitted together with the file. In the general case, where sender and recipient cannot communicate such information, a stricter degree of *TEI conformance* will be required for loss-free interchange. The rules defining such strict conformance to the Guidelines are given in some detail in chapter 28 ('Conformance') on p. 611. The *interchange format* defined there requires that an electronic text:

1. adhere to the SGML declaration and the SGML document type declarations defined in these Guidelines, unless modified or extended as described in chapter 29 ('Modifying the TEI DTD') on p. 619. These SGML constructs are further discussed in chapter 2 ('A Gentle Introduction to SGML ') on p. 15.
2. provide external documentation as described in chapter 27 ('Tag Set Documentation') on p. 601 for all elements not defined in these Guidelines, specifying a formal name (generic identifier) and a corresponding full natural-language name, describing its meaning and usage, specifying its legal content and also any attributes it may use.
3. adhere to the requirements of the TEI header in providing bibliographic identification of the text and description of the encoding practices used (as described in chapter 5 ('The TEI Header') on p. 77).

Note that the interchange format makes no formal restriction on the character set to be used in interchange, as this will depend on the medium of interchange and the local character sets in use by sender and receiver. For further information, refer to chapter 30 ('Rules for Interchange ') on p. 627.

1.2.2.3 Use for Local Processing

Machine-readable text can be manipulated in many ways; some users:

- edit texts (e.g. word processors, syntax-directed editors)
- edit, display, and link texts in hypertext systems

- format and print texts using desktop publishing systems, or batch-oriented formatting programs
- load texts into free-text retrieval databases or conventional databases
- unload texts from databases as search results or for export to other software
- search texts for words or phrases
- perform content analysis on texts
- collate texts for critical editions
- scan texts for automatic indexing or similar purposes
- parse texts linguistically
- analyze texts stylistically
- scan verse texts metrically
- link text and images

These applications cover a wide range of likely uses but are by no means exhaustive. The aim has been to make the TEI Guidelines useful for encoding the same texts for different purposes. We have avoided anything which would restrict the use of the text for other applications. We have also tried not to omit anything essential to any single application.

1.3 Historical Background

The Text Encoding Initiative grew out of a planning conference sponsored by the Association for Computers and the Humanities (ACH) and funded by the U.S. National Endowment for the Humanities (NEH), which was held at Vassar College in November 1987. At this conference some thirty representatives of text archives, scholarly societies, and research projects met to discuss the feasibility of a standard encoding scheme and to make recommendations for its scope, structure, content, and drafting. During the conference, the Association for Computational Linguistics and the Association for Literary and Linguistic Computing agreed to join ACH as sponsors of a project to develop the Guidelines. The outcome of the conference was this set of principles, which determined the further course of the project.

1. The guidelines are intended to provide a standard format for data interchange in humanities research.
2. The guidelines are also intended to suggest principles for the encoding of texts in the same format.
3. The guidelines should
   (a) define a recommended syntax for the format,
   (b) define a metalanguage for the description of text-encoding schemes,
   (c) describe the new format and representative existing schemes both in that metalanguage and in prose.
4. The guidelines should propose sets of coding conventions suited for various applications.
5. The guidelines should include a minimal set of conventions for encoding new texts in the format.
6. The guidelines are to be drafted by committees on
   (a) text documentation
   (b) text representation
   (c) text interpretation and analysis
   (d) metalanguage definition and description of existing and proposed schemes,

   coordinated by a steering committee of representatives of the principal sponsoring organizations.
7. Compatibility with existing standards will be maintained as far as possible.
8. A number of large text archives have agreed in principle to support the guidelines in their function as an interchange format. We encourage funding agencies to support development of tools to facilitate this interchange.
9. Conversion of existing machine-readable texts to the new format involves the translation of their conventions into the syntax of the new format. No requirements will be made for the

addition of information not already coded in the texts.

In the course of the work, some of these goals assumed greater, some lesser importance; some proved easier, some harder to achieve. The document in hand does define a standard form for the interchange of textual material, and adumbrate principles for the creation of new electronic texts. The only metalanguage used, however, is that of SGML, and no formal definitions are given of other common encoding schemes. These Guidelines do define a minimal set of conventions for text encoding (i.e. those SGML elements classed as recommended or required), though few researchers will be satisfied to encode *only* what is required or recommended here, since the set of required and recommended SGML elements is rather small. This document does not, however, define — at least not explicitly — "sets of coding conventions suited for various applications", since consensus on suitable conventions for different applications proved elusive; this remains a goal for future work.

1.3.1 Origin and Development of the TEI

The Text Encoding Initiative proper began in June 1988 with funding from the NEH, soon followed by further funding from the Commission of the European Communities, the Andrew W. Mellon Foundation, and the Social Science and Humanities Research Council of Canada. Four working committees, composed of distinguished scholars and researchers from both Europe and North America, were named to deal with problems of text documentation (resulting largely in chapter 5 ('The TEI Header') on p. 77), text representation, text analysis and interpretation (together responsible for most of what has become parts II, III, and IV), and metalanguage and syntax issues (largely responsible for part VI). A first draft version (1.0) of the Guidelines was distributed in July 1990 under the title *Guidelines for the Encoding and Interchange of Machine-Readable Texts*, with the TEI document number TEI P1. With minor changes and corrections, this version was reprinted as version 1.1 in November 1990. Extensive public comment and further work on areas not covered in version 1 resulted in the drafting of a revised version, TEI P2, distribution of which began in April 1992. This version includes substantial amounts of new material, resulting from work carried out by several specialist working groups, set up in 1990 and 1991 to propose extensions and revisions to the text of P1. The overall organization, both of the draft itself and of the scheme it describes, was entirely revised and reorganized in response to public comment on the first draft. In June, 1993, the Advisory Board of the Text Encoding Initiative met to review the current state of the Guidelines, and recommended the formal publication of the work done to that time. The present version of the TEI Guidelines, TEI P3, represents a further revision of all chapters published under the document number TEI P2, and the addition of further chapters. Although it will be subject to revision and amendment on the basis of practical experience and public discussion, this version of the Guidelines is published without the label 'draft', and marks the conclusion of the initial development work.

1.3.2 Future Developments

Work on areas still not satisfactorily covered in this manual will continue, and resulting recommendations will be issued as supplements to the published Guidelines. Work is expected to continue in at least the following areas:

- linguistic description and grammatical annotation
- historical analysis and interpretation
- base tag sets for further document types
- manuscript analysis and physical description of text

The encoding recommended by this document may be used without fear that future versions of the TEI scheme will be inconsistent with it in fundamental ways. The TEI will be sensitive, in revising these Guidelines, to the possible problems which revision might pose for those who are already using this draft. Wherever consistent with the long-term goals of the project, consistency with this version will be preserved in future revisions.

Chapter 2

A Gentle Introduction to SGML

The encoding scheme defined by these Guidelines is formulated as an application of a system known as the Standard Generalized Markup Language (SGML). [1] SGML is an international standard for the definition of device-independent, system-independent methods of representing texts in electronic form. This chapter presents a brief tutorial guide to its main features, for those readers who have not encountered it before. For a more technical account of TEI practice in using the SGML standard, see chapter 28 ('Conformance') on p. 611; for a more technical description of the subset of SGML used by the TEI encoding scheme, see chapter 39 ('Formal Grammar for the TEI-Interchange-Format Subset of SGML') on p. 993.

SGML is an international standard for the description of marked-up electronic text. More exactly, SGML is a *metalanguage*, that is, a means of formally describing a language, in this case, a *markup language*. Before going any further we should define these terms.

Historically, the word *markup* has been used to describe annotation or other marks within a text intended to instruct a compositor or typist how a particular passage should be printed or laid out. Examples include wavy underlining to indicate boldface, special symbols for passages to be omitted or printed in a particular font and so forth. As the formatting and printing of texts was automated, the term was extended to cover all sorts of special *markup codes* inserted into electronic texts to govern formatting, printing, or other processing.

Generalizing from that sense, we define markup, or (synonymously) *encoding*, as any means of making explicit an interpretation of a text. At a banal level, all printed texts are encoded in this sense: punctuation marks, use of capitalization, disposition of letters around the page, even the spaces between words, might be regarded as a kind of markup, the function of which is to help the human reader determine where one word ends and another begins, or how to identify gross structural features such as headings or simple syntactic units such as dependent clauses or sentences. Encoding a text for computer processing is in principle, like transcribing a manuscript from *scriptio continua*, a process of making explicit what is conjectural or implicit, a process of directing the user as to how the content of the text should be interpreted.

By *markup language* we mean a set of markup conventions used together for encoding texts. A markup language must specify what markup is allowed, what markup is required, how markup is to be distinguished from text, and what the markup means. SGML provides the means for doing the first three; documentation such as these Guidelines is required for the last.

The present chapter attempts to give an informal introduction—much less formal than the standard itself—to those parts of SGML of which a proper understanding is necessary to make best use of these Guidelines.

2.1 What's Special about SGML?

[1] International Organization for Standardization, *ISO 8879: Information processing—Text and office systems—Standard Generalized Markup Language (SGML)*, ([Geneva]: ISO, 1986).

There are three characteristics of SGML which distinguish it from other markup languages: its emphasis on descriptive rather than procedural markup; its *document type* concept; and its independence of any one system for representing the script in which a text is written. These three aspects are discussed briefly below, and then in more depth in sections 2.3 ('SGML Structures') on p. 17 and 2.7 ('SGML Entities ') on p. 29.

2.1.1 Descriptive Markup

A descriptive markup system uses markup codes which simply provide names to categorize parts of a document. Markup codes such as `<para>` or `\end{list}` simply identify a portion of a document and assert of it that "the following item is a paragraph," or "this is the end of the most recently begun list," etc. By contrast, a procedural markup system defines what processing is to be carried out at particular points in a document: "call procedure PARA with parameters 1, b and x here" or "move the left margin 2 quads left, move the right margin 2 quads right, skip down one line, and go to the new left margin," etc. In SGML, the instructions needed to process a document for some particular purpose (for example, to format it) are sharply distinguished from the descriptive markup which occurs within the document. Usually, they are collected outside the document in separate procedures or programs.

With descriptive instead of procedural markup the same document can readily be processed by many different pieces of software, each of which can apply different processing instructions to those parts of it which are considered relevant. For example, a content analysis program might disregard entirely the footnotes embedded in an annotated text, while a formatting program might extract and collect them all together for printing at the end of each chapter. Different sorts of processing instructions can be associated with the same parts of the file. For example, one program might extract names of persons and places from a document to create an index or database, while another, operating on the same text, might print names of persons and places in a distinctive typeface.

2.1.2 Types of Document

Secondly, SGML introduces the notion of a *document type*, and hence a *document type definition* (DTD). Documents are regarded as having types, just as other objects processed by computers do. The type of a document is formally defined by its constituent parts and their structure. The definition of a report, for example, might be that it consisted of a title and possibly an author, followed by an abstract and a sequence of one or more paragraphs. Anything lacking a title, according to this formal definition, would not formally be a report, and neither would a sequence of paragraphs followed by an abstract, whatever other report-like characteristics these might have for the human reader.

If documents are of known types, a special purpose program (called a *parser*) can be used to process a document claiming to be of a particular type and check that all the elements required for that document type are indeed present and correctly ordered. More significantly, different documents of the same type can be processed in a uniform way. Programs can be written which take advantage of the knowledge encapsulated in the document structure information, and which can thus behave in a more intelligent fashion.

2.1.3 Data Independence

A basic design goal of SGML was to ensure that documents encoded according to its provisions should be transportable from one hardware and software environment to another without loss of information. The two features discussed so far both address this requirement at an abstract level; the third feature addresses it at the level of the strings of bytes (characters) of which documents are composed. SGML provides a general purpose mechanism for *string substitution*, that is, a simple machine-independent way of stating that a particular string of characters in the document should be replaced by some other string when the document is processed. One obvious application for this mechanism is to ensure consistency of nomenclature; another, more significant one, is to

counter the notorious inability of different computer systems to understand each other's character sets, or of any one system to provide all the graphic characters needed for a particular application, by providing descriptive mappings for non-portable characters. The strings defined by this string-substitution mechanism are called *entities* and they are discussed below in section 2.7 ('SGML Entities ') on p. 29.

2.2 Textual Structure

A text is not an undifferentiated sequence of words, much less of bytes. For different purposes, it may be divided into many different units, of different types or sizes. A prose text such as this one might be divided into sections, chapters, paragraphs, and sentences. A verse text might be divided into cantos, stanzas, and lines. Once printed, sequences of prose and verse might be divided into volumes, gatherings, and pages.

Structural units of this kind are most often used to identify specific locations or reference points within a text ("the third sentence of the second paragraph in chapter ten"; "canto 10, line 1234"; "page 412," etc.) but they may also be used to subdivide a text into meaningful fragments for analytic purposes ("is the average sentence length of section 2 different from that of section 5?" "how many paragraphs separate each occurrence of the word 'nature'?" "how many pages?"). Other structural units are more clearly analytic, in that they characterize a section of a text. A dramatic text might regard each speech by a different character as a unit of one kind, and stage directions or pieces of action as units of another kind. Such an analysis is less useful for locating parts of the text ("the 93rd speech by Horatio in Act 2") than for facilitating comparisons between the words used by one character and those of another, or those used by the same character at different points of the play.

In a prose text one might similarly wish to regard as units of different types passages in direct or indirect speech, passages employing different stylistic registers (narrative, polemic, commentary, argument, etc.), passages of different authorship and so forth. And for certain types of analysis (most notably textual criticism) the physical appearance of one particular printed or manuscript source may be of importance: paradoxically, one may wish to use descriptive markup to describe presentational features such as typeface, line breaks, use of white space and so forth.

These textual structures overlap with each other in complex and unpredictable ways. Particularly when dealing with texts as instantiated by paper technology, the reader needs to be aware of both the physical organization of the book and the logical structure of the work it contains. Many great works (Sterne's *Tristram Shandy* for example) cannot be fully appreciated without an awareness of the interplay between narrative units (such as chapters or paragraphs) and page divisions. For many types of research, it is the interplay between different levels of analysis which is crucial: the extent to which syntactic structure and narrative structure mesh, or fail to mesh, for example, or the extent to which phonological structures reflect morphology.

2.3 SGML Structures

This section describes the simple and consistent mechanism for the markup or identification of structural textual units which is provided by SGML. It also describes the methods SGML provides for the expression of rules defining how combinations of such units can meaningfully occur in any text.

2.3.1 Elements

The technical term used in the SGML standard for a textual unit, viewed as a structural component, is *element*. Different types of elements are given different names, but SGML provides no way of expressing the meaning of a particular type of element, other than its relationship to other element types. That is, all one can say about an element called (for instance) <blort> is that

instances of it may (or may not) occur within elements of type <farble>, and that it may (or may not) be decomposed into elements of type <blortette>. It should be stressed that the SGML standard is entirely unconcerned with the semantics of textual elements: these are application dependent.[2] It is up to the creators of SGML conformant tag sets (such as these Guidelines) to choose intelligible names for the elements they identify and to document their proper use in text markup. That is one purpose of this document. From the need to choose element names indicative of function comes the technical term for the name of an element type, which is *generic identifier*, or GI.

Within a marked up text (a *document instance*), each element must be explicitly marked or tagged in some way. The standard provides for a variety of different ways of doing this, the most commonly used being to insert a tag at the beginning of the element (a *start-tag*) and another at its end (an *end-tag*). The start- and end-tag pair are used to bracket off the element occurrences within the running text, in rather the same way as different types of parentheses or quotation marks are used in conventional punctuation. For example, a quotation element in a text might be tagged as follows:

```
...  Rosalind's remarks <quote>This is the silliest stuff
 that ere I heard of!</quote> clearly indicate ...
```

As this example shows, a start-tag takes the form <name>, where the opening angle bracket indicates the start of the start-tag, "name" is the generic identifier of the element which is being delimited, and the closing angle bracket indicates the end of a tag. An end-tag takes an identical form, except that the opening angle bracket is followed by a solidus (slash) character, so that the corresponding end-tag would be </name>. [3]

2.3.2 Content Models: An Example

An element may be *empty*, that is, it may have no content at all, or it may contain simple text. More usually, however, elements of one type will be *embedded* (contained entirely) within elements of a different type.

To illustrate this, we will consider a very simple structural model. Let us assume that we wish to identify within an anthology only poems, their titles, and the stanzas and lines of which they are composed. In SGML terms, our document type is the *anthology*, and it consists of a series of *poem*s. Each poem has embedded within it one element, a title, and several occurrences of another, a stanza, each stanza having embedded within it a number of line elements. Fully marked up, a text conforming to this model might appear as follows: [4]

```
<anthology>
  <poem><title>The SICK ROSE</title>
     <stanza>
          <line>O Rose thou art sick.</line>
          <line>The invisible worm,</line>
          <line>That flies in the night</line>
          <line>In the howling storm:</line>
     </stanza>
     <stanza>
          <line>Has found out thy bed</line>
          <line>Of crimson joy:</line>
          <line>And his dark secret love</line>
          <line>Does thy life destroy.</line>
     </stanza>
  </poem>

          <!-- more poems go here    -->
```

[2] Work is currently going on in the standards community to create (using SGML syntax) a definition of a standard "document style semantics and specification language" or DSSSL.

[3] The actual characters used for the delimiting characters (the angle brackets, exclamation mark and solidus) may be redefined, but it is conventional to use the characters used in this description.

[4] The example is taken from William Blake's *Songs of innocence and experience* (1794). The markup is designed for illustrative purposes and is not TEI-conformant.

```
</anthology>
```

It should be stressed that this example does *not* use the same names as are proposed for corresponding elements elsewhere in these Guidelines: the above is *not* a valid TEI document. It will however serve as an introduction to the basic notions of SGML. White space and line breaks have been added to the example for the sake of visual clarity only; they have no particular significance in the SGML encoding itself. Also, the line

```
<!-- more poems go here    -->
```

is an SGML *comment* and is not treated as part of the text.

This example makes no assumptions about the rules governing, for example, whether or not a title can appear in places other than preceding the first stanza, or whether lines can appear which are not included in a stanza: that is why its markup appears so verbose. In such cases, the beginning and end of every element must be explicitly marked, because there are no identifiable rules about which elements can appear where. In practice, however, rules can usually be formulated to reduce the need for so much tagging. For example, considering our greatly over-simplified model of a poem, we could state the following rules:

1. An anthology contains a number of poems and nothing else.
2. A poem always has a single title element which precedes the first stanza and contains no other elements.
3. Apart from the title, a poem consists only of stanzas.
4. Stanzas consist only of lines and every line is contained by a stanza.
5. Nothing can follow a stanza except another stanza or the end of a poem.
6. Nothing can follow a line except another line or the start of a new stanza.

From these rules, it may be inferred that we do not need to mark the ends of stanzas or lines explicitly. From rule 2 it follows that we do not need to mark the end of the title—it is implied by the start of the first stanza. Similarly, from rules 3 and 1 it follows that we need not mark the end of the poem: since poems cannot occur within poems but must occur within anthologies, the end of a poem is implied by the start of the next poem, or by the end of the anthology. Applying these simplifications, we could mark up the same poem as follows:

```
<anthology>
     <poem><title>The SICK ROSE
     <stanza>
          <line>O Rose thou art sick.
          <line>The invisible worm,
          <line>That flies in the night
          <line>In the howling storm:
     <stanza>
          <line>Has found out thy bed
          <line>Of crimson joy:
          <line>And his dark secret love
          <line>Does thy life destroy.

     <poem>
          <!-- more poems go here    -->

</anthology>
```

The ability to use rules stating which elements can be nested within others to simplify markup is a very important characteristic of SGML. Before considering these rules further, you may wish to consider how text marked up in the form above could be processed by a computer for very many different purposes. A simple indexing program could extract only the relevant text elements in order to make a list of titles, or of words used in the poem text; a simple formatting program could insert blank lines between stanzas, perhaps indenting the first line of each, or inserting a stanza number. Different parts of each poem could be typeset in different ways. A more ambitious

analytic program could relate the use of punctuation marks to stanzaic and metrical divisions. [5] Scholars wishing to see the implications of changing the stanza or line divisions chosen by the editor of this poem can do so simply by altering the position of the tags. And of course, the text as presented above can be transported from one computer to another and processed by any program (or person) capable of making sense of the tags embedded within it with no need for the sort of transformations and translations needed to move word processor files around.

2.4 Defining SGML Document Structures: The DTD

Rules such as those described above are the first stage in the creation of a formal specification for the structure of an SGML document, or *document type definition*, usually abbreviated to *DTD*. In creating a DTD, the document designer may be as lax or as restrictive as the occasion warrants. A balance must be struck between the convenience of following simple rules and the complexity of handling real texts. This is particularly the case when the rules being defined relate to texts which already exist: the designer may have only the haziest of notions as to an ancient text's original purpose or meaning and hence find it very difficult to specify consistent rules about its structure. On the other hand, where a new text is being prepared to an exact specification, for example for entry into a textual database of some kind, the more precisely stated the rules, the better they can be enforced. Even in the case where an existing text is being marked up, it may be beneficial to define a restrictive set of rules relating to one particular view or hypothesis about the text—if only as a means of testing the usefulness of that view or hypothesis. It is important to remember that every document type definition is an interpretation of a text. There is no single DTD which encompasses any kind of absolute truth about a text, although it may be convenient to privilege some DTDs above others for particular types of analysis.

At present, SGML is most widely used in environments where uniformity of document structure is a major desideratum. In the production of technical documentation, for example, it is of major importance that sections and subsections should be properly nested, that cross references should be properly resolved and so forth. In such situations, documents are seen as raw material to match against pre-defined sets of rules. As discussed above, however, the use of simple rules can also greatly simplify the task of tagging accurately elements of less rigidly constrained texts. By making these rules explicit, the scholar reduces his or her own burdens in marking up and verifying the electronic text, while also being forced to make explicit an interpretation of the structure and significant particularities of the text being encoded.

2.4.1 An Example DTD

A DTD is expressed in SGML as a set of declarative statements, using a simple syntax defined in the standard. For our simple model of a poem, the following declarations would be appropriate:

```
<!ELEMENT anthology      - -  (poem+)>
<!ELEMENT poem           - -  (title?, stanza+)>
<!ELEMENT title          - O  (#PCDATA) >
<!ELEMENT stanza         - O  (line+)   >
<!ELEMENT line           O O  (#PCDATA) >
```

These five lines are examples of formal SGML element declarations. A declaration, like an element, is delimited by angle brackets; the first character following the opening bracket must be an exclamation mark, followed immediately by one of a small set of SGML-defined keywords, specifying the kind of object being declared. The five declarations above are all of the same type: each begins with an `ELEMENT` keyword, indicating that it declares an element, in the technical sense defined above. Each consists of three parts: a name or group of names, two characters specifying *minimization rules*, and a *content model*. Each of these parts is discussed further below. Components of the declaration are separated by white space, that is one or more blanks, tabs or newlines.

[5] Note that this simple example has not addressed the problem of marking elements such as sentences explicitly; the implications of this are discussed below in section 2.5.2 ('Concurrent Structures ') on p. 24.

The first part of each declaration above gives the generic identifier of the element which is being declared, for example 'poem', 'title', etc. It is possible to declare several elements in one statement, as discussed below.

2.4.2 Minimization Rules

The second part of the declaration specifies what are called *minimization rules* for the element concerned. These rules determine whether or not start- and end-tags must be present in every occurrence of the element concerned. They take the form of a pair of characters, separated by white space, the first of which relates to the start-tag, and the second to the end-tag. In either case, either a hyphen or a letter O (for "omissible" or "optional") must be given; the hyphen indicating that the tag must be present, and the letter O that it may be omitted. Thus, in this example, every element except <line> must have a start-tag. Only the <poem> and <anthology> elements must have end-tags as well.

2.4.3 Content Model

The third part of each declaration, enclosed in parentheses, is called the *content model* of the element, because it specifies what element occurrences may legitimately contain. Contents are specified either in terms of other elements or using special reserved words. There are several such reserved words, of which by far the most commonly encountered is #PCDATA, as in this example. This is an abbreviation for 'parsed character data,' and it means that the element being defined may contain any valid character data. If an SGML declaration is thought of as a structure like a family tree, with a single ancestor at the top (in our case, this would be <anthology>), then almost always, following the branches of the tree downwards (for example, from <anthology> to <poem> to <stanza> to <line> and <title>) will lead eventually to #PCDATA. In our example, <title> and <line> are so defined. Since their content models say #PCDATA only and name no embedded elements, they may not contain any embedded elements.

2.4.4 Occurrence Indicators

The declaration for <stanza> in the example above states that a stanza consists of one or more lines. It uses an *occurrence indicator* (the plus sign) to indicate how many times the element named in its content model may occur. There are three occurrence indicators in the SGML syntax, conventionally represented by the plus sign, the question mark, and the asterisk or star. [6] The plus sign means that there may be one or more occurrences of the element concerned; the question mark means that there may be at most one and possibly no occurrence; the star means that the element concerned may either be absent or appear one or more times. Thus, if the content model for <stanza> were (LINE*), stanzas with no lines would be possible as well as those with more than one line. If it were (LINE?), again empty stanzas would be countenanced, but no stanza could have more than a single line. The declaration for <poem> in the example above thus states that a <poem> cannot have more than one title, but may have none, and that it must have at least one <stanza> and may have several.

2.4.5 Group Connectors

The content model (TITLE?, STANZA+) contains more than one component, and thus needs additionally to specify the order in which these elements (<title> and <stanza>) may appear. This ordering is determined by the *group connector* (the comma) used between its components. There are three possible group connectors, conventionally represented by comma, vertical bar, and ampersand. [7] The comma means that the components it connects must both appear in the

[6] Like the delimiters, these are assigned formal names by the standard and may be redefined with an appropriate SGML declaration.

[7] What are here called "group connectors" are referred to by the SGML standard simply as "connectors"; the longer term is preferred here to stress the fact that these connectors are used only in SGML model groups and name groups. Like the delimiters and the occurrence indicators, group connectors are assigned formal names by the standard and may be redefined with an appropriate SGML declaration.

order specified by the content model. The ampersand indicates that the components it connects must both appear but may appear in any order. The vertical bar indicates that only one of the components it connects may appear. If the comma in this example were replaced by an ampersand, a title could appear either before the stanzas of a <poem> or at the end (but not between stanzas). If it were replaced by a vertical bar, then a <poem> would consist of either a title or just stanzas—but not both!

2.4.6 Model Groups

In our example so far, the components of each content model have been either single elements or #PCDATA. It is quite permissible however to define content models in which the components are lists of elements, combined by group connectors. Such lists, known as *model groups*, may also be modified by occurrence indicators and themselves combined by group connectors. To demonstrate these facilities, let us now expand our example to include non-stanzaic types of verse. For the sake of demonstration, we will categorize poems as one of *stanzaic*, *couplets*, or *blank* (or *stichic*). A blank-verse poem consists simply of lines (we ignore the possibility of verse paragraphs for the moment) [8] so no additional elements need be defined for it. A couplet is defined as a <line1> followed by a <line2>.

```
<!ELEMENT couplet O O (line1, line2) >
```

The elements <line1> and <line2> (which are distinguished to enable studies of rhyme scheme, for example) have exactly the same content model as the existing <line> element. They can therefore share the same declaration. In this situation, it is convenient to supply a *name group* as the first component of a single element declaration, rather than give a series of declarations differing only in the names used. A name group is a list of GIs connected by any group connector and enclosed in parentheses, as follows:

```
<!ELEMENT (line | line1 | line2) O O (#PCDATA) >
```

The declaration for the <poem> element can now be changed to include all three possibilities:

```
<!ELEMENT poem - O (title?, (stanza+ | couplet+ | line+) ) >
```

That is, a poem consists of an optional title, followed by one or several stanzas, or one or several couplets, or one or several lines. Note the difference between this definition and the following:

```
<!ELEMENT poem - O (title?, (stanza | couplet | line)+ ) >
```

The second version, by applying the occurrence indicator to the group rather than to each element within it, would allow for a single poem to contain a mixture of stanzas, couplets or blank verse.

Quite complex models can easily be built up in this way, to match the structural complexity of many types of text. As a further example, consider the case of stanzaic verse in which a refrain or chorus appears. A refrain may be composed of repetitions of the line element, or it may simply be text, not divided into verse lines. A refrain can appear at the start of a poem only, or as an optional addition following each stanza. This could be expressed by a content model such as the following:

```
<!ELEMENT refrain - - (#PCDATA | line+)>
<!ELEMENT poem    - O (title?,
                       ( (line+)
                       | (refrain?, (stanza, refrain?)+ ) )) >
```

That is, a poem consists of an optional title, followed by either a sequence of lines, or an un-named group, which starts with an optional refrain, followed by one of more occurrences of another group, each member of which is composed of a stanza followed by an optional refrain. A sequence such as 'refrain - stanza - stanza - refrain' follows this pattern, as does the sequence

[8] It will not have escaped the astute reader that the fact that verse paragraphs need not start on a line boundary seriously complicates the issue; see further section 2.5.2 ('Concurrent Structures ') on p. 24.

'stanza - refrain - stanza - refrain'. The sequence 'refrain - refrain - stanza - stanza' does not, however, and neither does the sequence "stanza - refrain - refrain - stanza." Among other conditions made explicit by this content model are the requirements that at least one stanza must appear in a poem, if it is not composed simply of lines, and that if there is both a title and a stanza they must appear in that order.

2.5 Complicating the Issue: More on Element Declarations

In the simple cases described so far, it has been assumed that one can identify the immediate constituents of every element defined in a textual structure. A poem consists of stanzas, and an anthology consists of poems. Stanzas do not float around unattached to poems or combined into some other unrelated element; a poem cannot contain an anthology. All the elements of a given document type may be arranged into a hierarchic structure, arranged like a family tree with a single ancestor at the top and many children (mostly the elements containing #PCDATA) at the bottom. This gross simplification turns out to be surprisingly effective for a large number of purposes. It is not however adequate for the full complexity of real textual structures. In particular, it does not cater for the case of more or less freely floating elements that can appear at almost any hierarchic level in the structure, and it does not cater for the case where different elements overlap or several different trees may be identified in the same document. To deal with the first case, SGML provides the *exception* mechanism; to deal with the second, SGML permits the definition of "concurrent" document structures.

2.5.1 Exceptions to the Content Model

In most documents, there will be some elements that can occur at any level of its structure. Annotations, for example, might be attached to the whole of a poem, to a stanza, to a line of a stanza or to a single word within it. In a textual critical edition, the same might be true of variant readings. In this simple case, the complexity of adding an annotation element as an optional component of every content model is not particularly onerous; in a more realistically complex model perhaps containing some ten or twenty levels such an approach can become much more difficult.

To cope with this, SGML allows for any content model to be further modified by means of an *exception* list. There are two types of exception: *inclusions*, that is, additional elements that can be included at any point in the model group or any of its constituent elements; and *exclusions*, that is, elements that cannot be included within the current model.

To extend our declarations further to allow for annotations and variant readings, which we will assume can appear anywhere within the text of a poem, we first need to add declarations for these two elements:

```
<!ELEMENT (note | variant) - - (#PCDATA)>
```

The note and variant elements must have both start- and end-tags, since they can appear anywhere. Rather than add them to the content model for each type of poem, we can add them in the form of an inclusion list to the poem element, which now reads:

```
<!ELEMENT poem - O (title?, (stanza+ | couplet+ | line+) )
                                        +(note | variant) >
```

The plus sign at the start of the (NOTE | VARIANT) name list indicates that this is an inclusion exception. With this addition, notes or variants can appear at any point in the content of a poem element—even those (such as <title>) for which we have defined a content model of #PCDATA. They can thus also appear within notes or variants!

If we wanted for some reason to prevent notes or variants appearing within titles, we could add an exclusion exception to the declaration for <title> above:

```
<!ELEMENT title  - O  (#PCDATA)  -(note | variant) >
```

The minus sign at the start of the (NOTE | VARIANT) name list indicates that this is an exclusion exception. With this addition, notes and variants will be prohibited from appearing within titles, notwithstanding their potential inclusion implied by the previous addition to the content model for <poem>.

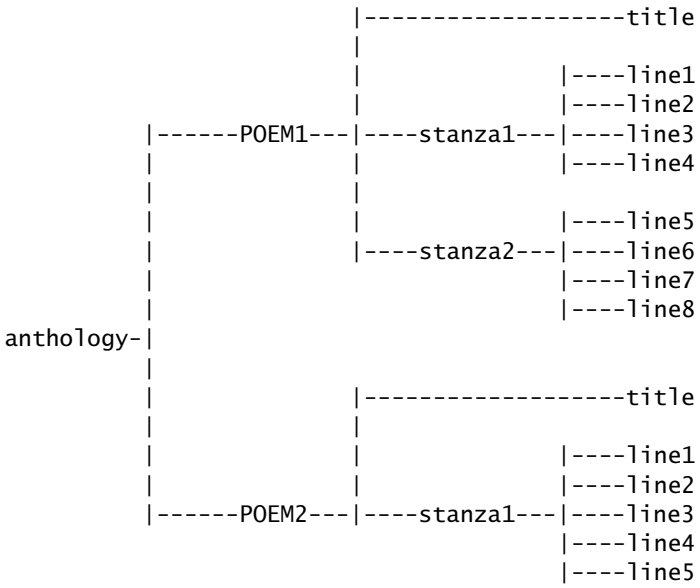
In the same way, we could prevent notes and variants from nesting within notes and variants by modifying the definition above to read

```
<!ELEMENT (note | variant) - - (#PCDATA  -(note | variant) >
```

The meticulous reader will note that this precludes both variants within notes and notes within variants. Inclusion and exclusion exceptions should be used with care as their ramifications may not be immediately apparent.

2.5.2 Concurrent Structures

All the structures we have so far discussed have been simply hierarchic: that is, at every level of the tree, each node is entirely contained by a parent node. The figure below represents the structure of a document conforming to the simple DTD we have so far defined as a tree (drawn on its side through exigencies of space). We have already seen how Blake's poem can be divided into a title and two stanzas, each of four lines. In this diagram, we add a second poem, consisting of one stanza and a title, to make up an instance of an anthology:



Clearly, there are many such trees that might be drawn to describe the structure of this or other anthologies. Some of them might be representable as further subdivisions of this tree: for example, we might subdivide the lines into individual words, since no word crosses a line boundary. But equally clearly there are many other trees that might be drawn which do *not* fit within this tree. We might, for example, be interested in syntactic structures — which rarely respect the formal boundaries of verse. Or, to take a simpler example, we might want to represent the pagination of different editions of the same text.

One way of doing this would be to group the lines and titles of our current model into pages. A declaration for such an element is simple enough:

```
<!ELEMENT page - - ((title?, line+)+)   >
```

That is, a page consists of one or more unnamed groups, each of which contains an optional title, followed by a sequence of lines. (Note, incidentally, that this model prohibits a title appearing on its own at the foot of a page). However, simply inserting the element <page> into the hierarchy already defined is not as easy as it might seem. Some poems are longer than a single page, and other pages contain more than one poem. We cannot therefore insert the element <page> between <anthology> and <poem> in the hierarchy, nor can it go between <poem> and <stanza>, nor yet in both places at once! What is needed is the ability to create a separate

hierarchy, with the same elements at the bottom (the stanzas, lines and titles), but combined into a different superstructure. This is the ability which the CONCUR feature of SGML gives.

A separate document type definition must be created for each hierarchic tree into which the text is to be structured. The definition we have so far built up for the anthology looks, in full, like this:

```
<!DOCTYPE anthology [
<!ELEMENT anthology      - -  (poem+)             >
<!ELEMENT poem           - -  (title?, stanza+)   >
<!ELEMENT stanza         - O  (line+)             >
<!ELEMENT (title | line) - O  (#PCDATA)           >
]>
```

As this example shows, the name of a document type must always be the same as the name of the largest element in it, that is the element at the top of the hierarchy. The syntax used is discussed further below (see section 2.9.2 ('The DTD') on p. 32). Let us now add to this declaration a second definition for a concurrent document type, which we will call a paged anthology, or <p.anth> for short:

```
<!DOCTYPE p.anth [
<!ELEMENT p.anth         - -  (page+)               >
<!ELEMENT page           - -  ((title?, line+)+)    >
<!ELEMENT (title|line)   - O  (#PCDATA)             >
]>
```

We have now defined two different ways of looking at the same basic text—the PCDATA components grouped by both these document type definitions into lines or titles. In one view, the lines are grouped into stanzas and poems; in the other they are grouped into pages only. Notice that it is exactly the same text which is visible in both views: the two hierarchies simply allow us to arrange it in two different ways.

To mark up the two views, it will be necessary to indicate which hierarchy each element belongs to. This is done by including the name of the document type (the view) within parentheses immediately before the identifier concerned, inside both start- and end-tags. Thus, pages (which are only visible in the <p.anth> document type) must be tagged with a <(p.anth)page> tag at their start and a </(p.anth)page> at their end. In the same way, as poems and stanzas appear only in the <anthology> document type, they must now be tagged using <(anthology)poem> and <(anthology)stanza> tags respectively. For the line and title elements, however, which appear in both hierarchies, no document type specification need be given: any tag containing only a name is assumed to mark an element present in every active document type.

As a simple example, let us assume that Blake's poem appears in some paged anthology, with the page break occurring half way through the first stanza. The poem might then be marked up as follows:

```
<(anthology)anthology>
<(p.anth)p.anth>
<(p.anth)page>

<!--      other titles and lines on this page here -->

     <(anthology)poem><title>The SICK ROSE
     <(anthology)stanza>
          <line>O Rose thou art sick.
          <line>The invisible worm,
</(p.anth)page>
<(p.anth)page>
          <line>That flies in the night
          <line>In the howling storm:
     <(anthology)stanza>
          <line>Has found out thy bed
          <line>Of crimson joy:
          <line>And his dark secret love
          <line>Does thy life destroy.
```

```
        </(anthology)poem>

    <!--      rest of material on this page here    -->
    </(p.anth)page>

    </(p.anth)p.anth)
    </(anthology)anthology>
```

It is now possible to select only the elements concerned with a particular view from the text, even though both are represented in the tagging. A processor concerned only with the pagination will see only those elements whose tags include the P.ANTH specification, or which have no specification at all. A processor concerned only with the ANTHOLOGY view of things will not see the page breaks. And a processor concerned to inter-relate the two views can do so unambiguously.

A note of caution is appropriate: CONCUR is an optional feature of SGML, and not all available SGML software systems support it, while those which do, do not always do so according to the letter of the standard. For that reason, if for no other, wherever these Guidelines have identified a potential application of CONCUR, they also invariably suggest alternative methods as well. For fuller discussion of these issues, see chapter 31 ('Multiple Hierarchies') on p. 633.

Note also that we cannot introduce a new element, a page number for example, into the <p.anth> document type, since there is no existing data in the <anthology> document type which could be fitted into it. One way of adding that extra information is discussed in the next section.

2.6 Attributes

In the SGML context, the word 'attribute', like some other words, has a specific technical sense. It is used to describe information which is in some sense descriptive of a specific element occurrence but not regarded as part of its content. For example, you might wish to add a status attribute to occurrences of some elements in a document to indicate their degree of reliability, or to add an identifier attribute so that you could refer to particular element occurrences from elsewhere within a document. Attributes are useful in precisely such circumstances.

Although different elements may have attributes with the same name, (for example, in the TEI scheme, every element is defined as having an id attribute), they are always regarded as different, and may have different values assigned to them. If an element has been defined as having attributes, the attribute values are supplied in the document instance as *attribute-value pairs* inside the start-tag for the element occurrence. An end-tag may not contain an attribute-value specification, since it would be redundant.

For example

```
<poem id=P1 status="draft"> ... </poem>
```

The <poem> element has been defined as having two attributes: id and status. For the instance of a <poem> in this example, represented here by an ellipsis, the id attribute has the value P1 and the status attribute has the value draft. An SGML processor can use the values of the attributes in any way it chooses; for example, a formatter might print a poem element which has the status attribute set to draft in a different way from one with the same attribute set to revised; another processor might use the same attribute to determine whether or not poem elements are to be processed at all. The id attribute is a slightly special case in that, by convention, it is always used to supply a unique value to identify a particular element occurrence, which can be used for cross reference purposes, as discussed further below.

Like elements, attributes are declared in the SGML document type declaration, using rather similar syntax. As well as specifying its name and the element to which it is to be attached, it is possible to specify (within limits) what kind of value is acceptable for an attribute and a default value.

The following declarations could be used to define the two attributes we have specified above for the <poem> element:

```
<!ATTLIST poem
          id       ID                              #IMPLIED
          status   (draft | revised | published)   draft       >
```

The declaration begins with the symbol ATTLIST, which introduces an *attribute list specification*. The first part of this specifies the element (or elements) concerned. In our example, attributes have been declared only for the <poem> element. If several elements share the same attributes, they may all be defined in a single declaration; just as with element declarations, several names may be given in a parenthesized list. Following this name (or list of names), is a series of rows, one for each attribute being declared, each containing three parts. These specify the name of the attribute, the type of value it takes, and a default value respectively.

Attribute names (id and status in this example) are subject to the same restrictions as other names in SGML; they need not be unique across the whole DTD, however, but only within the list of attributes for a given element.

The second part of an attribute specification can take one of two forms, both illustrated above. The first case uses one of a number of special keywords to declare what kind of value an attribute may take. In the example above, the special keyword ID is used to indicate that the attribute ID will be used to supply a unique identifying value for each poem instance (see further the discussion below). Among other possible SGML keywords are

CDATA The attribute value may contain any valid character data; tags may be included in the value, but they will not be recognized by the SGML parser, and will not be processed as tags normally are

IDREF The attribute value must contain a pointer to some other element (see further the discussion of ID below)

NMTOKEN The attribute value is a *name token*, that is, (more or less) any string of alphanumeric characters

NUMBER The attribute value is composed only of numerals

In the example above, a list of the possible values for the status attribute has been supplied. This means that a parser can check that no <poem> is defined for which the status attribute does not have one of draft, revised, or published as its value. Alternatively, if the declared value had been either CDATA or NAME, a parser would have accepted almost any string of characters (status=awful or status=12345678 if it had been a NMTOKEN; status="anything goes" or status = "well, ALMOST anything" if it were CDATA). Sometimes, of course, the set of possible values cannot be pre-defined. Where it can, as in this case, it is generally better to do so.

The last piece of each information in each attribute definition specifies how a parser should interpret the absence of the attribute concerned. This can be done by supplying one of the special keywords listed below, or (as in this case) by supplying a specific value which is then regarded as the value for every element which does not supply a value for the attribute concerned. Using the example above, if a poem is simply tagged <poem>, the parser will treat it exactly as if it were tagged <poem status=draft>. Alternatively, one of the following keywords may be used to specify a default value for an attribute:

#REQUIRED A value must be specified.

#IMPLIED A value need not be supplied (as in the case of ID above).

#CURRENT If no value is supplied in this element occurrence, the last specified value should be used.

For example, if the attribute definition above were rewritten as

```
<!ATTLIST poem
          id       ID                              #IMPLIED
          status   (draft | revised | published)   #CURRENT     >
```

then poems which appear in the anthology simply tagged <poem> would be treated as if they had the same status as the preceding poem. If the keyword were #REQUIRED rather than #CURRENT, the parser would report such poems as erroneously tagged, as it would if any value other than draft, published, or revised were supplied. The use of #CURRENT implies that whatever value is specified for this attribute on the first poem will apply to all subsequent

poems, until altered by a new value. Only the status of the first poem need therefore be supplied, if all are the same.

It is sometimes necessary to refer to an occurrence of one textual element from within another, an obvious example being phrases such as "see note 6" or "as discussed in chapter 5." When a text is being produced the actual numbers associated with the notes or chapters may not be certain. If we are using descriptive markup, such things as page or chapter numbers, being entirely matters of presentation, will not in any case be present in the marked up text: they will be assigned by whatever processor is operating on the text (and may indeed differ in different applications). SGML therefore provides a special mechanism by which any element occurrence may be given a special identifier, a kind of label, which may be used to refer to it from anywhere else within the same text. The cross-reference itself is regarded as an element occurrence of a specific kind, which must also be declared in the DTD. In each case, the identifying label (which may be arbitrary) is supplied as the value of a special attribute.

Suppose, for example, we wish to include a reference within the notes on one poem that refers to another poem. We will first need to provide some way of attaching a label to each poem: this is done by defining an attribute for the <poem> element, as suggested above.

```
<!ATTLIST poem
          id       ID     #IMPLIED >
```

Here we define an attribute id, the value of which must be of type ID. It is not required that any attribute of type ID have the name id as well; it is however a useful convention almost universally observed. Note that not every poem need carry an id attribute and the parser may safely ignore the lack of one in those which do not. Only poems to which we intend to refer need use this attribute; for each such poem we should now include in its start-tag some unique identifier, for example:

```
<POEM ID=Rose>
     Text of poem with identifier 'ROSE'
</POEM>

<POEM ID=P40>
     Text of poem with identifier 'P40'
</POEM>

<POEM>
     This poem has no identifier
</POEM>
```

Next we need to define a new element for the cross reference itself. This will not have any content—it is only a pointer—but it has an attribute, the value of which will be the identifier of the element pointed at. This is achieved by the following declarations:

```
<!ELEMENT poemref - O EMPTY                  >
<!ATTLIST poemref     target IDREF #REQUIRED >
```

The <poemref> element needs no end-tag because it has no content. It has a single attribute called target. The value of this attribute must be of type IDREF (the keyword used for cross reference pointers of this type) and it must be supplied.

With these declarations in force, we can now encode a reference to the poem with id Rose as follows:

```
Blake's poem on the sick rose <POEMREF TARGET=Rose> ...
```

When an SGML parser encounters this empty element it will simply check that an element exists with the identifier Rose. Different SGML processors could take any number of additional actions: a formatter might construct an exact page and line reference for the location of the poem in the current document and insert it, or just quote the poem's title or first lines. A hypertext style processor might use this element as a signal to activate a link to the poem being referred to. The purpose of the SGML markup is simply to indicate that a cross reference exists: it does not determine what the processor is to do with it.

2.7 SGML Entities

The aspects of SGML discussed so far are all concerned with the markup of structural elements within a document. SGML also provides a simple and flexible method of encoding and naming arbitrary parts of the actual content of a document in a portable way. In SGML the word *entity* has a special sense: it means a named part of a marked up document, irrespective of any structural considerations. An entity might be a string of characters or a whole file of text. To include it in a document, we use a construction known as an *entity reference*. For example, the following declaration

```
<!ENTITY tei "Text Encoding Initiative">
```

defines an entity whose name is *tei* and whose value is the string "Text Encoding Initiative." [9]

This is an instance of an *entity declaration*, which declares an *internal entity*. The following declaration, by contrast, declares a *system entity*:

```
<!ENTITY ChapTwo SYSTEM "sgmlmkup.txt">
```

This defines a system entity whose name is *ChapTwo* and whose value is the text associated with the system identifier — in this case, the system identifier is the name of an operating system file and the replacement text of the entity is the contents of the file.

Once an entity has been declared, it may be referenced anywhere within a document. This is done by supplying its name prefixed with the ampersand character and followed by the semicolon. The semicolon may be omitted if the entity reference is followed by a space or record end.

When an SGML parser encounters such an *entity reference,* it immediately substitutes the value declared for the entity name. Thus, the passage "The work of the &tei has only just begun" will be interpreted by an SGML processor exactly as if it read "The work of the Text Encoding Initiative has only just begun". In the case of a system entity, it is, of course, the contents of the operating system file which are substituted, so that the passage "The following text has been suppressed: &ChapTwo;" will be expanded to include the whole of whatever the system finds in the file *sgmlmkup.txt*.[10]

This obviously saves typing, and simplifies the task of maintaining consistency in a set of documents. If the printing of a complex document is to be done at many sites, the document body itself might use an entity reference, such as `&site;`, wherever the name of the site is required. Different entity declarations could then be added at different sites to supply the appropriate string to be substituted for this name, with no need to change the text of the document itself.

This *string substitution* mechanism has many other applications. It can be used to circumvent the notorious inadequacies of many computer systems for representing the full range of graphic characters needed for the display of modern English (let alone the requirements of other modern scripts or of ancient languages). So-called "special characters" not directly accessible from the keyboard (or if accessible not correctly translated when transmitted) may be represented by an entity reference.

Suppose, for example, that we wish to encode the use of ligatures in early printed texts. The ligatured form of 'ct' might be distinguished from the non-ligatured form by encoding it as `&ctlig;` rather than `ct.` Other special typographic features such as leafstops or rules could equally well be represented by mnemonic entity references in the text. When processing such texts, an entity declaration would be added giving the desired representation for such textual elements. If, for example, ligatured letters are of no interest, we would simply add a declaration such as

```
<!ENTITY ctlig "ct" >
```

[9] By convention case is significant in entity names, unlike element names.

[10] Strictly speaking, SGML does not require system entities to be files; they can in principle be any data source available to the SGML processor: files, results of database queries, results of calls to system functions — anything at all. It is simpler, however, when first learning SGML, to think of system entities as referring to files, and this discussion therefore ignores the other possibilities. All existing SGML processors do support the use of system entities to refer to files; fewer support the other possible uses of system entities.

and the distinction present in the source document would be removed. If, on the other hand, a formatting program capable of representing ligatured characters is to be used, we might replace the entity declaration to give whatever sequence of characters such a program requires as the expansion.

A list of entity declarations is known as an *entity set*. Standard entity sets are provided for use with most SGML processors, in which the names used will normally be taken from the lists of such names published as an annex to the SGML standard and elsewhere, as mentioned above.

The replacement values given in an entity declaration are, of course, highly system dependent. If the characters to be used in them cannot be typed in directly, SGML provides a mechanism to specify characters by their numeric values, known as *character references*. A character reference is distinguished from other characters in the replacement string by the fact that it begins with a special symbol, conventionally the sequence '&#', and ends with the normal semicolon. For example, if the formatter to be used represents the ligatured form of ct by the characters c and t prefixed by the character with decimal value 102, the entity declaration would read:

```
<!ENTITY ctlig "&#102;ct" >
```

Note that character references will generally not make sense if transferred to another hardware or software environment: for this reason, their use is only recommended in situations like this.

Useful though the entity reference mechanism is for dealing with occasional departures from the expected character set, no one would consider using it to encode extended passages, such as quotations in Greek or Russian in an English text. In such situations, different mechanisms are appropriate. These are discussed elsewhere in these Guidelines (see chapter 4 ('Characters and Character Sets') on p. 71).

A special form of entities, *parameter entities*, may be used within SGML markup declarations; these differ from the entities discussed above (which technically are known as *general entities*) in two ways:

- Parameter entities are used *only* within SGML markup declarations; with some special exceptions which will not be discussed here, they will normally not be found within the document itself.
- Parameter entities are delimited by percent sign and semicolon, rather than by ampersand and semicolon.

Declarations for parameter entities take the same form as those for general entities, but insert a percent sign between the keyword `ENTITY` and the name of the entity itself. White space (blanks, tabs, or line breaks) must occur on both sides of the percent sign. An internal parameter entity named *TEI.prose*, with an expansion of `INCLUDE`, and an external parameter entity named *TEI.extensions.dtd*, which refers to the system file *mystuff.dtd*, could be declared thus:[11]

```
<!ENTITY % TEI.prose 'INCLUDE'>
<!ENTITY % TEI.extensions.dtd SYSTEM 'mystuff.dtd'>
```

The TEI document type definition makes extensive use of parameter entities to control the selection of different tag sets and to make it easier to modify the TEI DTD. Numerous examples of their use may thus be found in chapter 3 ('Structure of the TEI Document Type Definition') on p. 35.

2.8 Marked Sections

It is occasionally convenient to mark some portion of a text for special treatment by the SGML parser. Certain portions of legal boilerplate, for example, might need to be included or omitted systematically, depending on the state or country in which the document was intended to be valid. (Thus the statement "Liability is limited to $50,000." might need to be included in Delaware, but excluded in Maryland.) Technical manuals for related products might share a great deal of

[11] Such entity declarations might be used in extending the TEI base tag set for prose using the declarations found in *mystuff.dtd*.

information but differ in some details; it might be convenient to maintain all the information for the entire set of related products in a single document, selecting at display or print time only those portions relevant to one specific product. (Thus, a discussion of how to change the oil in a car might use the same text for most steps, but offer different advice on removing the carburetor, depending on the specific engine model in question.)

SGML provides the *marked section* construct to handle such practical requirements of document production. In general, as the examples above are intended to suggest, it is more obviously useful in the production of new texts than in the encoding of pre-existing texts. Most users of the TEI encoding scheme will never need to use marked sections, and may wish to skip the remainder of this discussion. The TEI DTD makes extensive use of marked sections, however, and this section should be read and understood carefully by anyone wishing to follow in detail the discussions in chapter 3 ('Structure of the TEI Document Type Definition') on p. 35.

The "special processing" offered for marked sections in SGML can be of several types, each associated with one of the following keywords:

INCLUDE The marked section should be included in the document and processed normally.

IGNORE The marked section should be ignored entirely; if the SGML application program produces output from the document, the marked section will be excluded from the document.

CDATA The marked section may contain strings of characters which look like SGML tags or entity references, but which should not be recognized as such by the SGML parser. (These Guidelines use such CDATA marked sections to enclose the examples of SGML tagging.)

RCDATA The marked section may contain strings of characters which look like SGML tags, but which should not be recognized as such by the SGML parser; entity references, on the other hand, may be present and should be recognized and expanded as normal.

TEMP The passage included in the marked section is a temporary part of the document; the marked section is used primarily to indicate its location, so that it can be removed or revised conveniently later.

When a marked section occurs in the text, it is preceded by a *marked-section start* string, which contains one or more keywords from the list above; its end is marked by a *marked-section close* string. The second and last lines of the following example are the start and close of a marked section to be ignored:

```
In such cases, the bank will reimburse the customer for all losses.
<![ IGNORE [
Liability is limited to $50,000.
]]>
```

Of the marked section keywords, the most important for understanding the TEI DTD are `INCLUDE` and `IGNORE`; these can be used to include and exclude portions of a document — or a DTD — selectively, so as to adjust it to relevant circumstances (e.g. to allow a user to select portions of the DTD relevant to the document in question).

The literal keywords `INCLUDE` and `IGNORE`, however, are not much use in adjusting a DTD or a document to a user's requirements, however. (To change the text above to include the excluded sentence, for example, a user would have to edit the text manually and change `IGNORE` to `INCLUDE`. It might be thought just as easy to add and delete the sentence manually.) But the keywords need not be given as literal values; they can be represented by a parameter entity reference. In a document with many sentences which should be included only in Maryland, for example, each such sentence can be included in a marked section whose keyword is represented by a reference to a parameter entity named *Maryland*. The earlier example would then be:

```
In such cases, the bank will reimburse the customer for all losses.
<![ %Maryland; [
Liability is limited to $50,000.
]]>
```

When the entity *Maryland* is defined as `IGNORE`, the marked sections so marked will all be excluded. If the definition is changed to the following, the marked sections will be included in the document:

```
<!ENTITY % Maryland 'INCLUDE'>
```

When parameter entities are used in this way to control marked sections in a DTD, the external DTD file normally contains a default declaration. If the user wishes to override the default (as by including the Maryland sections), adding an appropriate declaration to the DTD subset suffices to override the default.[12]

The examples of parameter entity declarations at the end of the preceding section can now be better understood. The declarations

```
<!ENTITY % TEI.prose 'INCLUDE'>
<!ENTITY % TEI.extensions.dtd SYSTEM 'mystuff.dtd'>
```

have the effect of *including* in the DTD all the sections marked as relevant to prose, since in the external DTD files such sections are all included in marked sections controlled by the parameter entity *TEI.prose*. They also override the default declaration of *TEI.extensions.dtd* (which declares this entity as an empty string), so as to include the file *mystuff.dtd* in the DTD.

2.9 Putting It All Together

An SGML conformant document has a number of parts, not all of which have been discussed in this chapter, and many of which the user of these Guidelines may safely ignore. For completeness, the following summary of how the parts are inter-related may however be found useful.

An SGML document consists of an SGML *prolog* and a *document instance*. The prolog contains an *SGML declaration* (described below) and a *document type definition*, which contains element and entity declarations such as those described above. Different software systems may provide different ways of associating the document instance with the prolog; in some cases, for example, the prolog may be "hard-wired" into the software used, so that it is completely invisible to the user.

2.9.1 The SGML Declaration

The SGML declaration specifies basic facts about the dialect of SGML being used such as the character set, the codes used for SGML delimiters, the length of identifiers, etc. Its content for TEI-conformant document types is discussed further in chapters 39 ('Formal Grammar for the TEI-Interchange-Format Subset of SGML') on p. 993 and 28 ('Conformance') on p. 611. Normally the SGML declaration will be held in the form of compiled tables by the SGML processor and will thus be invisible to the user.

2.9.2 The DTD

The document type definition specifies the document type definition against which the document instance is to be validated. Like the SGML declaration it may be held in the form of compiled tables within the SGML processor, or associated with it in some way which is invisible to the user, or requires only that the name of the document type be specified before the document is validated.

At its simplest the document type definition consists simply of a base document type definition (possibly also one or more concurrent document type definitions) which is prefixed to the document instance. For example:

```
<!DOCTYPE my.dtd [
    <!-- all declarations for MY.DTD go here -->
    ...
]>
<my.dtd>
    This is an instance of a MY.DTD type document
</my.dtd>
```

[12] This is so because the declarations in the DTD subset are read before those in the external DTD file, and the first declaration of a given entity is the one which counts. This was described briefly in section 2.7 ('SGML Entities ') on p. 29.

More usually, the document type definition will be held in a separate file and invoked by reference, as follows:

```
<!DOCTYPE tei.2 system "tei2.dtd" [
]>
<tei.2>
     This is an instance of an unmodified TEI type document
</tei.2>
```

Here, the text of the TEI.2 document type definition is not given explicitly, but the SGML processor is told that it may be read from the file with the system identifier given in quotation marks. The square brackets may still be supplied, as in this example, even though they enclose nothing.

The part enclosed by square brackets is known as the *document type declaration subset* or "DTD subset". Its purpose is to specify any modification to be made to the DTD being invoked, thus:

```
<!DOCTYPE tei.2 SYSTEM "tei2.dtd" [
     <!ENTITY tla "Three Letter Acronym">
     <!ELEMENT my.tag - - (#PCDATA)>
     <!-- any other special-purpose declarations or
          re-definitions go in here -->
]>
<tei.2>
     This is an instance of a modified TEI.2 type document,
     which may contain <my.tag>my special tags</my.tag> and
     references to my usual entities such as &tla;.
</tei.2>
```

In this case, the document type definition in force includes first the contents of the DTD subset, and then the contents of the file specified after the keyword SYSTEM. The order is important, because in SGML only the first declaration of an entity counts. In the above example, therefore, the declaration of the entity *tla* in the DTD subset would take precedence over any declaration of the same entity in the file *tei2.dtd*. It is perfectly legal SGML for entities to be declared twice; this is the usual method for allowing user modification of SGML DTDs. (Elements, by contrast, may not be declared more than once; if a declaration for <my.tag> were contained in file *tei.dtd*, the SGML parser would signal an error.) Combining and extending the TEI document type definitions is discussed further in chapter 3 ('Structure of the TEI Document Type Definition') on p. 35.

2.9.3 The Document Instance

The document instance is the content of the document itself. It contains only text, markup and general entity references, and thus may not contain any new declarations. A convenient way of building up large documents in a modular fashion might be to use the DTD subset to declare entities for the individual pieces or modules, thus:

```
<!DOCTYPE tei.2 [
     <!ENTITY chap1 system "chap1.txt">
     <!ENTITY chap2 system "chap2.txt">
     <!ENTITY chap3 "-- not yet written --">
]>
<tei.2>
<teiHeader> ... </teiHeader>
<text>
    <front> ... </front>
    <body>
     &chap1;
     &chap2;
     &chap3;
     ...
</body>
</text>
```

```
</tei.2>
```

In this example, the DTD contained in file *tei2.dtd* has been extended by entity declarations for each chapter of the work. The first two are system entities referring to the file in which the text of particular chapters is to be found; the third a dummy, indicating that the text does not yet exist (alternatively, an entity with a null value could be used). In the document instance, the entity references &chap1; etc. will be resolved by the parser to give the required contents. The chapter files themselves will not, of course, contain any element, attribute list, or entity declarations—just tagged text.

2.10 Using SGML

A variety of software is available to assist in the tasks of creating, validating and processing SGML documents. Only a few basic types can be described here. At the heart of most such software is an SGML *parser*: that is, a piece of software which can take a document type definition and generate from it a software system capable of validating any document invoking that DTD. Output from a parser, at its simplest, is just "yes" (the document instance is valid) or "no" (it is not). Most parsers will however also produce a new version of the document instance in *canonical form* (typically with all end-tags supplied and entity references resolved) or formatted according to user specifications. This form can then be used by other pieces of software (loosely or tightly coupled with the parser) to provide additional functions, such as structured editing, formatting and database management.

A *structured editor* is a kind of intelligent word-processor. It can use information extracted from a processed DTD to prompt the user with information about which elements are required at different points in a document as the document is being created. It can also greatly simplify the task of preparing a document, for example by inserting tags automatically.

A *formatter* operates on a tagged document instance to produce a printed form of it. Many typographic distinctions, such as the use of particular typefaces or sizes, are intimately related to structural distinctions, and formatters can thus usefully take advantage of descriptive markup. It is also possible to define the tagging structure expected by a formatting program in SGML terms, as a concurrent document structure.

Text-oriented database management systems typically use inverted file indexes to point into documents, or subdivisions of them. A search can be made for an occurrence of some word or word pattern within a document or within a subdivision of one. Meaningful subdivisions of input documents will of course be closely related to the subdivisions specified using descriptive markup. It is thus simple for textual database systems to take advantage of SGML-tagged documents. Much research work is also currently going into ways of extending the capabilities of existing (non-text) database systems to take advantage of the structuring information made explicit by SGML markup.

Hypertext systems improve on other methods of handling text by supporting associative links within and across documents. Again, the basic building block needed for such systems is also a basic building block of SGML markup: the ability to identify and to link together individual document elements comes free as a part of the SGML way of doing things. By tagging links explicitly, rather than using proprietary software, developers of hypertexts can be sure that the resources they create will continue to be useful. To load an SGML document into a hypertext system requires only a processor which can correctly interpret SGML tags such as those discussed in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331.

Chapter 3

Structure of the TEI Document Type Definition

This chapter describes the overall structure of the encoding scheme defined by these Guidelines. It introduces the conceptual framework within which the following chapters are to be understood, and describes the technical means by which that conceptual framework is implemented in SGML. It assumes some familiarity with SGML; see chapter 2 ('A Gentle Introduction to SGML ') on p. 15.

The TEI encoding scheme consists of number of modules or *DTD fragments* which we refer to below as *tag sets*. Selected tag sets may be combined in many different ways, according to principles described in this chapter, within the framework of the TEI *main DTD*. Auxiliary tag sets are also defined for specific purposes independent of the TEI main DTD.

The DTD fragments from which the main TEI DTD is constructed may be classified as follows:

- core DTD fragments
- base DTD fragments
- additional DTD fragments

The first two sections of this chapter discuss these distinctions and list the specific tag sets included in each category. Section 3.3 ('Invocation of the TEI DTD') on p. 39 describes how to invoke the TEI document type declaration, and how to specify which of the various base tag sets and optional additional tag sets are used in a document.

The *global attributes*, characteristics postulated of every element or tag in the encoding scheme, are defined in section 3.5 ('Global Attributes') on p. 42.

The remainder of the chapter contains a more technical description of the SGML mechanisms used to implement the encoding scheme. It may be skipped at a first reading, but a proper understanding of the topics addressed here is essential for anyone planning to modify or extend the TEI encoding scheme in any way (see also chapter 29 ('Modifying the TEI DTD') on p. 619), and also highly desirable for those wishing to take full advantage of its modular nature. The structure of the main TEI DTD file itself is outlined in section 3.6 ('The TEI2.DTD File') on p. 45. The *element classes* used to define smaller groups of elements and their characteristics are described in section 3.7 ('Element Classes') on p. 51. Both global attributes and element classes are implemented using SGML *parameter entities*; various other uses of parameter entities in the TEI DTDs are discussed in section 3.8 ('Other Parameter Entities in TEI DTDs') on p. 65.

3.1 Main and Auxiliary DTDs

These Guidelines define a large number of SGML tags for marking up documents, all of which are formally defined within the *document type declaration* (*DTD*) files provided by the TEI and documented in the remainder of the present document. The tags are grouped into *tag sets* or

DTD fragments, each comprising a set of declarations for tags which belong together in some respect, typically related to their intended application area.

All tags used to transcribe documents are available for use within the *main DTD* of the TEI and are defined in Parts III and IV of these Guidelines. There are DTD fragments for prose and mixed matter, verse and verse collections, drama, dictionaries, analysis and interpretation of text, text criticism, etc. A full list, including the files in which they are defined, and the rules determining their selection and combination, is given in section 3.2 ('Core, Base, and Additional Tag Sets') on p. 36.

A number of *auxiliary DTDs* are also defined in these Guidelines. These are used for the encoding of ancillary descriptive information useful when processing electronic documents. Part V of these Guidelines describes several such auxiliary document types, specifically:

independent header for use with sets of TEI headers regarded as documents in their own right, for example by libraries or archives exchanging details of their holdings (see chapters 5 ('The TEI Header') on p. 77 and 24 ('The Independent Header') on p. 559).

writing system declaration used to define and document character sets or transliteration schemes (see chapters 4 ('Characters and Character Sets') on p. 71 and 25 ('Writing System Declaration') on p. 571).

feature system declaration used to define and document sets of analytic features (see chapters 16 ('Feature Structures') on p. 397 and 26 ('Feature System Declaration') on p. 589).

tag set declaration used to define and document descriptive documentation for TEI-conformant tag sets (see chapter 27 ('Tag Set Documentation') on p. 601).

An independent header typically describes the encoding of a specific document, but in the case of a planned corpus or collection, it may define a set of encoding practices common to all texts in the collection. The other auxiliary document types provide information likely to be relevant to many documents, rather than to individual documents.

When individual TEI documents are exchanged between sites, they should be accompanied by whatever auxiliary documents apply to them. When larger groups of documents are exchanged, the relevant auxiliary documents need be exchanged only once. For further information see chapter 30 ('Rules for Interchange ') on p. 627.

The DTD files containing these auxiliary DTDs are:

teishd2.dtd independent header
teiwsd2.dtd writing system declaration
teifsd2.dtd feature system declaration
teitsd2.dtd tag set declaration

Some of these auxiliary DTDs also make use of the core tag set defined as part of the main TEI DTD; this is described in the relevant chapters of part V.

3.2 Core, Base, and Additional Tag Sets

The main TEI DTD is constructed by selecting an appropriate combination of smaller tag sets, each containing some set of tags likely to be used together. These building blocks include:

core tag sets standard components of the TEI main DTD in all its forms; these are always included without any special action by the encoder;

base tag sets basic building blocks for specific text types; exactly one base must be selected by the encoder (unless one of the "combined" bases is used);

additional tag sets extra tags useful for particular purposes. All additional tag sets are compatible with all bases and with each other; an encoder may therefore add them to the selected base in any combination desired.

Each tag set is contained in one or more system files, which are defined by appropriate SGML *parameter entity declarations* and invoked as a unit by appropriate SGML *parameter entity references*. [1]

[1] A *parameter entity* is an SGML entity used only in markup declarations; references to parameter entities are delimited by a percent sign and a semicolon rather than the ampersand and colon used for *general entity* references. The entity

Several such declarations may be needed to invoke all parts of a given tag set, since as well as defining elements or attributes, a tag set may (for example) add new items to the set of global attributes or add classes to the system of element classes. Consistent naming principles are applied throughout the TEI scheme for these and other entities. Thus, assuming a tag set named *xxx*, the following parameter entities may be encountered:

TEI.xxx used to enable or disable tag set *xxx*; must have the value `INCLUDE` (tag set is enabled) or, by default, `IGNORE` (tag set not enabled).

TEI.xxx.ent refers to a system file containing any parameter entity declarations unique to tag set *xxx*.

TEI.xxx.dtd refers to a system file containing the element and attribute list declarations for tag set *xxx*.

a.xxx contains definitions of attributes which are to be added to the set of global attributes when tag set *xxx* is enabled.

m.comp.xxx if *xxx* is a base tag set, this contains a list of any component-level elements unique to it (for a definition of component-level elements, see section 3.7 ('Element Classes') on p. 51).

mix.xxx a special entity for use in defining the set of component-level elements when the mixed base tag set is in use.

gen.xxx a special entity for use in defining the set of component-level elements when the general base tag set is in use.

Few tag sets declare all of these entities; only those actually used are declared.

The interpretation of the parameter entity declarations, and the inclusion of the appropriate tag sets, are handled by a single "driver file" for the main TEI DTD. This file, *tei2.dtd*, is described in detail below in section 3.6 ('The TEI2.DTD File') on p. 45. The remainder of the present section identifies the files in which each tag set is contained, and the parameter entities associated with them.

3.2.1 The Core Tag Sets

Two "core" tag sets are always included in every invocation of the main TEI DTD. The tags and attributes that they contain are therefore available to any TEI document. The parameter entities used for this purpose, and the files they refer to, are:

TEI.core.dtd refers to the file *teicore2.dtd*, which declares the core tags defined in chapter 6 ('Elements Available in All TEI Documents') on p. 119

TEI.header.dtd refers to the file *teihdr2.dtd*, which declares the tags of the TEI header defined in chapter 5 ('The TEI Header') on p. 77

Together with these tag sets, part II also documents a tag set for default text structure and front and back matter. This tag set is embedded by the base tag set selected, and may vary with the base; it is therefore described in the next section.

3.2.2 The Base Tag Sets

The base tag sets are those which define the basic building blocks of different text types. The basic structures of verse (line, stanza, canto, etc.), for example, are not those of prose (paragraph, section, chapter, etc.), while dictionaries use yet another set of basic structures. Each base corresponds to one chapter of Part III of this document.

In general, exactly one base tag set must be selected for any TEI-conformant document. Errors will result if none, or more than one, is selected, because the same elements may be differently defined in different base tag sets. For documents which mingle structurally dissimilar elements and require elements from more than one base, however, either the *mixed base* or the *general base* may be used; see section 3.4 ('Combining TEI Base Tag Sets') on p. 40. These bases require the encoder to specify which of the other bases are to be combined.

TEI.core.ent, for example, would be referred to using the string `%TEI.core.ent;`. Parameter entities can also be used to control the inclusion or exclusion of *marked sections* of the document or DTD; the TEI DTD uses marked sections to handle the selection of different base and additional tag sets.

The encoder selects a base tag set by declaring the appropriate SGML parameter entity with the replacement text INCLUDE. To invoke the base tag set for prose, for example, the encoder must ensure that the DTD subset in the document contains the declaration:

```
<!ENTITY % TEI.prose 'INCLUDE' >
```

The entities used to select the different base tag sets, and the files containing the SGML declarations for each base, are listed below.

TEI.prose selects the base tag set for prose, contained in *teipros2.dtd*.
TEI.verse selects the base tag set for verse, contained in *teivers2.dtd* and *teivers2.ent*.
TEI.drama selects the base tag set for drama, contained in *teidram2.dtd* and *teidram2.ent*.
TEI.spoken selects the base tag set for transcriptions of spoken texts, contained in *teispok2.dtd* and *teispok2.ent*.
TEI.dictionaries selects the base tag set for print dictionaries, contained in *teidict2.dtd* and *teidict2.ent*.
TEI.terminology selects the base tag set for terminological data files, contained in *teiterm2.dtd*, *teiterm2.ent*, *teite2n.dtd*, and *teite2f.ent*.
TEI.general selects the generic mixed-mode base base tag set, contained in *teigen2.dtd*.
TEI.mixed selects the base tag set for free mixed-mode texts, contained in *teimix2.dtd*.

As shown in the list, each base tag set is normally contained in one or two system files: a required one (with the extension 'dtd') defining the elements in the tag set and their attributes, and an optional one (with the file extension 'ent') defining any global attributes or specialized element classes enabled by that tag set. The parameter entities for these files have the same name as the enabling parameter entity for the base, with the suffixes 'ent' and 'dtd' respectively: the prose base, for example, is enabled by declaring the parameter entity *TEI.prose* as INCLUDE; this in turn enables declarations of *TEI.prose.ent* and *TEI.prose.dtd* as the system files *teipros2.ent* and *teipros2.dtd*. For further details, see section 3.6 ('The TEI2.DTD File') on p. 45.

Most base tag sets (but not necessarily all) embed common definitions of text structure, front matter, and back matter, by referring to three standard parameter entities; these are:

TEI.structure.dtd refers to the file *teistr2.dtd*, with default definitions for <text>, , etc.
TEI.front.dtd refers to the file *teifron2.dtd*, with tags for front matter
TEI.back.dtd refers to the file *teiback2.dtd*, with tags for back matter

These default-structure tags are documented in chapter 7 ('Default Text Structure') on p. 183.

3.2.3 The Additional Tag Sets

The additional tag sets define optional tags required by different encoders for different types of analysis and processing; each corresponds to a chapter in part IV of this document. In any TEI encoding, any or all of these additional tag sets may be made available, as they are all compatible with each other and with every base tag set. They are invoked in the same way as base tag sets, by defining the appropriate parameter entity as INCLUDE; the relevant parameter entities, and the files containing the additional tag sets, are these:

TEI.linking embeds the files *teilink2.dtd* and *teilink2.ent*, with tags for linking, segmentation, and alignment (chapter 14 ('Linking, Segmentation, and Alignment') on p. 331)
TEI.analysis embeds the files *teiana2.dtd* and *teiana2.ent*, with tags for simple analytic mechanisms (chapter 15 ('Simple Analytic Mechanisms') on p. 381)
TEI.fs embeds the file *teifs2.dtd*, with tags for feature structure analysis (chapter 16 ('Feature Structures') on p. 397)
TEI.certainty embeds the file *teicert2.dtd*, with tags for indicating uncertainty and probability in the markup (chapter 17 ('Certainty and Responsibility') on p. 435)
TEI.transcr embeds the files *teitran2.dtd* and *teitran2.ent*, with tags for manuscripts, analytic bibliography, and transcription of primary sources (chapter 18 ('Transcription of Primary Sources') on p. 443)
TEI.textcrit embeds the files *teitc2.dtd* and *teitc2.ent*, with tags for critical editions (chapter 19 ('Critical Apparatus') on p. 467)

TEI.names.dates embeds the files *teind2.dtd* and *teind2.ent*, with specialized tags for names and dates (chapter 20 ('Names and Dates') on p. 487)

TEI.nets embeds the file *teinet2.dtd*, with tags for graphs, digraphs, trees, and other networks (chapter 21 ('Graphs, Networks, and Trees') on p. 505) — not to be confused with the graphics markup of *TEI.figures*

TEI.figures embeds the files *teifig2.dtd* and *teifig2.ent*, with tags for graphics, figures, illustrations, tables, and formulae (chapter 22 ('Tables, Formulae, and Graphics') on p. 523) — not to be confused with the graph-theoretic markup of *TEI.nets*

TEI.corpus embeds the file *teicorp2.dtd*, with tags for additional tags for language corpora (chapter 23 ('Language Corpora') on p. 537)

Like the base tag sets, the additional tag sets are each contained in one or two system files: a required one (with the file extension 'dtd') defining the elements in the tag set and their attributes, and an optional one (with the file extension 'ent') defining any global attributes or specialized element classes enabled by that tag set. The parameter entities for these files have the same name as the enabling parameter entity for the tag set, with the suffixes 'ent' and 'dtd' respectively: the additional tag set for linking, segmentation, and alignment, for example, is enabled by declaring the parameter entity *TEI.linking* as `INCLUDE`; this in turn enables declarations of *TEI.linking.ent* and *TEI.linking.dtd* as the system files *teilink2.ent* and *teilink2.dtd*.

3.2.4 User-Defined Tag Sets

As described in chapter 29 ('Modifying the TEI DTD') on p. 619, users may modify the markup language defined here by renaming elements, suppressing elements, adding new elements, or modifying element or attribute-list declarations. In general, local modifications will be most conveniently grouped into two files: one containing the local modifications to parameter entities used in the DTDs, and the other containing new or modified declarations of elements and their attributes. These files will be embedded in the TEI DTD if they are associated with the following two parameter entities:

TEI.extensions.ent local modifications to parameter entities

TEI.extensions.dtd declarations of new elements and modified declarations for existing elements

In some cases, users may wish to provide completely new base or additional tag sets, to be invoked in the same way as those defined in this document; such tag sets should also be divided into "entity files" and "DTD files" in the same way as the standard tag sets. Such modifications should be undertaken only with a thorough understanding of the interface among core, base, and additional tag sets as documented in the final sections of this chapter; see in particular section 3.6.2 ('Embedding Local Modifications') on p. 47.

Further recommendations for the creation of user-defined extension or modification are provided in chapters 29 ('Modifying the TEI DTD') on p. 619 and 28 ('Conformance') on p. 611.

3.3 Invocation of the TEI DTD

Like any other SGML document, a TEI document must begin with a document type definition (DTD). Local systems may allow the DTD to be implicit, but for interchange purposes it *must* be explicit. Because of its highly modular nature, it may in any case be desirable for the component parts of the TEI DTD to be made explicit even for local processing.

The simplest version of the TEI DTD names the main TEI DTD file as an external file, and specifies a single base tag set for use in the document, using the parameter entity names specified in section 3.2 ('Core, Base, and Additional Tag Sets') on p. 36. For example, a document using the base tag set for prose will begin with a document type declaration something like this:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
     <!ENTITY % TEI.prose 'INCLUDE' >
]>
```

A document using the base tag set for drama will define a different parameter entity:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
     <!ENTITY % TEI.drama 'INCLUDE' >
]>
```

If one or more of the *additional tag sets* described in Part IV are to be used, they are invoked in the same way as the base tag set. A document using the base tag set for prose, with the additional tag sets for text criticism and for linking, segmentation, and alignment, for example, will begin with a document type declaration something like this:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
<!-- TEI base tag set specified here: ... -->
     <!ENTITY % TEI.prose 'INCLUDE' >

<!-- TEI additional tag sets optionally specified here: ... -->
     <!ENTITY % TEI.textcrit 'INCLUDE' >
     <!ENTITY % TEI.linking 'INCLUDE' >
]>
```

If local modifications are used, they may be stored in separate files and pointed to using the parameter entities *TEI.extensions.ent* and *TEI.extensions.dtd*. If such local modifications are added to the example just given, this is the result:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
<!-- Local modifications to the TEI DTD declared here.  They
     will be embedded at an appropriate point in the main
     DTD. ... -->
     <!ENTITY % TEI.extensions.ent system 'project.ent' >
     <!ENTITY % TEI.extensions.dtd system 'project.dtd' >

<!-- TEI base tag set specified here: ... -->
     <!ENTITY % TEI.prose 'INCLUDE' >

<!-- TEI additional tag sets specified here: ... -->
     <!ENTITY % TEI.textcrit 'INCLUDE' >
     <!ENTITY % TEI.linking 'INCLUDE' >
]>
```

If the document requires tags which are defined in different base tag sets (e.g. prose and drama) or embeds smaller texts which use different base tag sets, then one of the mixed-type bases must be used. Their proper invocation is described below in section 3.4 ('Combining TEI Base Tag Sets') on p. 40.

3.4 Combining TEI Base Tag Sets

The TEI DTD has been designed to simplify the task of choosing an appropriate set of tags for the text in hand. The core tag set includes tags appropriate to the majority of simple tagging requirements for prose, verse and drama, irrespective of the base tag set chosen. For more detailed tagging, the encoder may choose the prose base for prose texts, the verse base for verse, and so on.

In discussing these base tag sets elsewhere in these Guidelines, it is generally assumed for clarity of exposition that a text will fall into one, not several, of these types. It is not uncommon, however, for a text to combine prose and verse, or other forms treated by the TEI as different bases. Examples include:

- when the text is a collection of other texts, which do not all use the same base: e.g. an anthology of prose, verse, and drama
- when the text contains other smaller, embedded texts: e.g. a poem or song included in a prose narrative

- when some sections of the text are written in one form, and others in a different form: e.g. a novel where some chapters are in prose, others take the form of dictionary entries and still others the form of scenes in a play
- when the text moves back and forth among forms not between sections but within a single section: e.g. mixed prose-and-verse forms like many pastorals or like some portions of the Poetic Edda

The TEI DTD provides the following mechanisms to handle these cases:

- a definition of a corpus or collection as a series of <tei.2> documents, sharing a common TEI header (see chapter 23 ('Language Corpora') on p. 537)
- a definition of composite texts which comprise front matter, a group or several possibly nested groups of collected texts, themselves possibly composite (see section 7.3 ('Groups of Texts') on p. 195)
- a notion of *embedded text* which allows one text to be embedded within another (that is, <text> is defined as a component-level element, as described briefly at the conclusion of section 7.3 ('Groups of Texts') on p. 195)

Whichever mechanism is adopted, if the whole of the resulting document is to be parseable by the main TEI DTD it may need to combine elements from different TEI base tag sets. Two special-purpose base tag sets are defined for this purpose:

- the *general* base, which allows different sections of a text to use different bases, but ensures that each section uses only one base
- the *mixed* base, which allows chunk- and inter-level elements from any base to mix within any text division

When either of these "combined" bases is used, the user must specify all of the other bases to be included in the mix as well as either the general or the mixed base. This is the only exception to the general rule that no more than one base tag set may be enabled in a TEI document. The following set of declarations for example allows for any mixture of the low level structural tags defined in the prose, drama and dictionary base tag sets:

```
<!DOCTYPE TEI.2 system 'tei.2' [
  <!ENTITY % TEI.mixed 'INCLUDE' >
  <!ENTITY % TEI.prose 'INCLUDE' >
  <!ENTITY % TEI.drama 'INCLUDE' >
  <!ENTITY % TEI.dictionaries 'INCLUDE' >
  <!-- Structurally, Moby Dick is not your
   everyday common or garden variety novel ... -->
]>
```

The following set of declarations has the same effect, but with the additional restriction that each text division (i.e. each member of the element class *divn*) must be homogenous with respect to the mixture of available bases. Because in a "general" base, each of the text may use a different base, the divisions of the text prefixed by this set of declarations will each be composed of elements taken solely from one of the prose, verse or dictionary base tag sets:

```
<!DOCTYPE TEI.2 system 'tei.2' [
  <!ENTITY % TEI.general 'INCLUDE' >
  <!ENTITY % TEI.prose 'INCLUDE' >
  <!ENTITY % TEI.drama 'INCLUDE' >
  <!ENTITY % TEI.dictionaries 'INCLUDE' >
]>
```

The actual DTD fragments for the combined bases do nothing but embed the default tag set for overall text structure. The mixed-base tag set is in file *teimix2.dtd*:

```
<!-- 3.4:  Mixed-Base Tag Set                                  -->
<!-- Text Encoding Initiative: Guidelines for Electronic      -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.    -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy    -->
<!-- in any form is granted, provided this notice is          -->
```

```
<!-- included in all copies.                                    -->

<!-- These materials may not be altered; modifications to       -->
<!-- these DTDs should be performed as specified in the         -->
<!-- Guidelines in chapter "Modifying the TEI DTD."             -->

<!-- These materials subject to revision. Current versions      -->
<!-- are available from the Text Encoding Initiative.           -->

<!ENTITY % TEI.structure.dtd system 'teistr2.dtd'               >
%TEI.structure.dtd;
```

The general-base tag set is in file *teigen2.dtd*:

```
<!-- 3.4:  General-Base Tag Set                                 -->
<!-- Text Encoding Initiative: Guidelines for Electronic        -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.      -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy      -->
<!-- in any form is granted, provided this notice is            -->
<!-- included in all copies.                                    -->

<!-- These materials may not be altered; modifications to       -->
<!-- these DTDs should be performed as specified in the         -->
<!-- Guidelines in chapter "Modifying the TEI DTD."             -->

<!-- These materials subject to revision. Current versions      -->
<!-- are available from the Text Encoding Initiative.           -->

<!ENTITY % TEI.structure.dtd system 'teistr2.dtd'               >
%TEI.structure.dtd;
```

3.5 Global Attributes

The following attributes are defined for every TEI element.[2]

id provides a unique identifier for the element bearing the ID value.

n gives a number (or other label) for an element, which is not necessarily unique within the document.

lang indicates the language of the element content, usually using a two- or three-letter code from ISO 639.

rend indicates how the element in question was rendered or presented in the source text.

Some tag sets (e.g. those for terminology, linking, and analysis) define other global attributes; these are documented in the appropriate chapters of Part III and Part IV. See also section 3.7.1 ('Classes Which Share Attributes') on p. 52.

An additional attribute, TEIform, is also defined for every TEI element. Unlike the other attributes defined for every element, TEIform is not defined by class *global* because its default value is different in every case and must be defined individually for each element.[3]

TEIform indicates the standard TEI name (generic identifier) for a given element.

Any TEI element may be given values for id, n, lang, rend, or TEIform, simply by specifying values for these attributes. The following two examples convey the same information about the text: that the material transcribed occurs within a <p> element (paragraph). They differ only in that the second provides an identifier for the paragraph, to which other elements (e.g. notes or hypertext links) can conveniently refer.

[2] More exactly, these are the attributes of the element class *global*, to which all elements belong; for further discussion of attribute classes and ways in which attributes may be inherited and over-ridden, see section 3.7.1 ('Classes Which Share Attributes') on p. 52.

[3] A dummy element class *TEIform* is defined in the reference section, solely for documentary purposes.

```
<p>If to do were as easy as to know what were
good to do, chapels had been churches and poor men's cottages
princes' palaces.  It is a good divine that follows his own
instructions ...</p>

<p id='MV1.2.5'>If to do were as easy as to know what were
good to do, chapels had been churches and poor men's cottages
princes' palaces.  It is a good divine that follows his own
instructions ...</p>
```

ID values must be legal SGML names; by default, this means they must begin with a letter from A to Z or a to z and contain no characters other than the letters A to Z or a to z, the digits 0 to 9, the full stop, and the hyphen. Furthermore, by default upper and lower case letters are not distinguished: thus, the strings 'a23' and 'A23' are identical, and may not be used to identify two distinct elements.

If two elements are given the same identifier, the SGML parser will signal a syntax error. The following example, therefore, is *not* valid:

```
<p id=page1><q>What's it going to be then, eh?</q></p>
<p id=page1>There was me, that is Alex, and my three droogs,
that is Pete, Georgie, and Dim, ... </p>
```

For a discussion of methods of providing unique identifiers for elements, see section 6.9.2 ('Creating New Reference Systems') on p. 157.

The n attribute allows identifying information (e.g. chapter numbers, etc.) to be encoded even if it would not be a legal id value. Its value may be any string of characters; typically it is a number or other similar enumerator or label. For example, the numbers given to the items of a numbered list may be recorded with the n attribute; this would make it possible to record errors in the numeration of the original, as in this list of chapters, transcribed from a faulty original in which the number 10 is used twice, and 11 is omitted:

```
<list type=ordered>
<item n=1>About These Guidelines
<item n=2>A Gentle Introduction to SGML
<!-- ... -->
<item n=9>Verse
<item n=10>Drama
<item n=10>Spoken Materials
<!-- sic:  original has '10' twice! -->
<item n=12>Printed Dictionaries
<!-- ... -->
</list>
```

The n attribute may also be used to record non-unique names associated with elements in a text, possibly together with a unique identifier as in the following examples:

```
<div type=chap n='One' id='TXT0101'>

<stanza n='xlii'>
```

The lang attribute indicates the language, writing system, and character set associated with a given element and all its contents. If it is not specified, the value is inherited from that of the immediately enclosing element. As a rule, therefore, it is simplest to specify the base language of the text on the <tei.2> element, and allow most elements to take the default value for lang; the language of an element then need be explicitly specified only for elements in languages other than the base language.

The following two encodings convey the same information about the language of the text, since in the first the lang attributes on the <emph> elements specify the same value as that on the parent <p> element, while in the second they inherit that value without specifying it.

```
<p lang=en> ... Both parties deprecated war, but one of
them would <emph lang=en>make</emph> war rather than let
the nation survive, and the other would <emph lang=en>accept
</emph> war rather than let it perish, and the war came.</p>
```

```
<p lang=en> ... Both parties deprecated war, but one of
them would <emph>make</emph> war rather than let
the nation survive, and the other would <emph>accept</emph>
war rather than let it perish, and the war came.</p>
```

In the following example, by contrast, the lang attribute on the <term> element must be given if we wish to record the fact that the technical terms used are Latin rather than English; no lang attribute is needed on the <q> element, by contrast, because it is in the same language as its parent. It is strongly recommended that all language shifts in the source be explicitly identified by use of the lang attribute, as described in chapter 4 ('Characters and Character Sets') on p. 71.

```
<p lang=en>The constitution declares <q>that no bill of
attainder or <term lang=la>ex post facto</term> law shall
be passed.</q> ... </p>
```

Formally, the lang attribute is an IDREF; a reference to the id value of a <language> element in the TEI header.[4] This means that each language used in the document should be declared in the TEI header using the <language> element defined in section 5.4.2 ('Language Usage') on p. 109.

The rend attribute is used to give information about the physical presentation of the text in the source. In the following example, it is used to indicate that both the emphasized word and the proper name are printed in italics:

```
<p> ... Their motives <emph rend='italics'>might</emph> be
pure and pious; but he was equally alarmed by his knowledge
of the ambitious <name rend='italics'>Bohemond</name>, and
his ignorance of the Transalpine chiefs: ...</p>
```

If all or most <emph> and <name> elements are rendered in the text by italics, it will be more convenient to register that fact in the TEI header once and for all and specify a rend value only for any elements which deviate from the usual rendition.

The contents of the rend attribute are free text. In any given project, encoders are advised to settle on a standard vocabulary with which to describe typographic or manuscript rendition of the text, and to document their usage of that vocabulary in the <rendition> element of the TEI header.

The TEIform attribute is used to allow application programs to handle TEI-encoded documents correctly even if some or all elements have been renamed. Most users can ignore this attribute entirely; it is only relevant when the TEI DTDs are modified. [5]

The default value of TEIform for any element is the generic identifier of that element, as described in this document. The value for <p> is 'p', the value for <div1> is 'div1', etc. When elements are renamed, as described in chapter 29 ('Modifying the TEI DTD') on p. 619, the declaration of TEIform is not modified. If <div1> is renamed <chapter>, for example, the default value of TEIform remains 'div1'. An application program which does not recognize the new generic identifier can check to see whether the attribute TEIform exists, and examine its value if it does to find out which TEI element, if any, is being used.

Modifications of DTDs, however, may involve more than simple renaming of elements: sometimes elements are given not just new names, but complete new definitions. In such cases, the TEIform attribute may be used to indicate the standard TEI element corresponding to the modified element. For example, if a local modification of a DTD renamed the <div1> element as <chapter> and also modified its formal declarations (e.g. to change its content model), then the TEIform attribute on the modified element should be given the default value div1, in order to indicate that the local <chapter> element is a modification of the standard TEI <div1>.

When new elements are introduced, they may be identified as specialized variants of existing TEI elements by giving them the appropriate default value for TEIform. For example, if a local element called <quatrain> were introduced, as a specialized variant of the <lg> (line group) element which must contain exactly four lines, then its declaration might give its TEIform as lg, to signify that a quatrain is a particular type of line group, thus:

[4] SGML validation checks that all IDREF values exist as id values on elements somewhere in the current SGML document. It is a requirement of the TEI scheme, not of SGML, that the lang attribute point to a <language> element.

[5] The TEIform attribute is based on the notion of *architectural forms* developed for HyTime (ISO 10744).

```
<!ELEMENT quatrain - O (l, l, l, l)            >
<!ATTLIST quatrain     %a.global
          TEIform      (lg)            'lg'    >
```

The formal definition of the global attributes is as follows:

```
<!-- 3.5:  Global attributes                                    -->

<!-- The global attributes are defined for every element in    -->
<!-- the TEI tag set; individual declarations may be           -->
<!-- overridden by local declarations for individual           -->
<!-- elements.                                                 -->

<!-- If the tag sets invoked by the user define extra global   -->
<!-- attributes (they do this in their .ent file), then they   -->
<!-- are inherited by GLOBAL; otherwise the parameter          -->
<!-- entities referred to expand to the empty string, as       -->
<!-- shown here.                                               -->


<!ENTITY % a.analysis ''                                        >
<!ENTITY % a.linking ''                                         >
<!ENTITY % a.terminology ''                                     >
<!ENTITY % a.global '        %a.analysis;
                             %a.linking;
                             %a.terminology;
          id                 ID                  #IMPLIED
          n                  CDATA               #IMPLIED
          lang               IDREF               %INHERITED
          rend               CDATA               #IMPLIED'     >

<!-- The TEIform attribute is also global, but is declared     -->
<!-- individually for each element, not in a parameter         -->
<!-- entity declaration.                                       -->

<!-- This fragment is used in sec. 3.7.3                       -->
```

3.6 The TEI2.DTD File

All TEI-encoded documents use the same top-level DTD file, which refers to a number of other DTD files, the exact set of other files referred to depending on which base and which additional tagsets are in use. The remainder of this chapter describes in some detail the organization and function of this file and those it embeds; it is necessarily of a rather technical and specialized nature.

The main TEI DTD is always invoked by specifying the file *tei2.dtd*. This file:

1. takes care of certain necessary preliminaries:
   (a) embeds any locally defined changes to the standard TEI parameter entities, so that local modifications can take precedence over default declarations;
   (b) declares TEI-specific keywords used in other declarations and declares default values of IGNORE for all the parameter entities used to select base and additional tag sets (see section 3.8.3 ('Parameter Entities for TEI Keywords') on p. 67);
   (c) declares parameter entities for TEI generic identifiers (by embedding the file *teigis2.ent*; see section 3.8.2 ('Parameter Entities for Element Generic Identifiers') on p. 66);
2. declares parameter entities for element classes, content models, and global attributes (by embedding *teiclas2.ent*; see section 3.7.3 ('The TEICLAS2.ENT File') on p. 55);
3. declares the top-level elements <TEI.2> and <teiCorpus.2>;

4. embeds DTD files containing local modifications (if any), the core tag sets, the base tag set, and the additional tag sets.

3.6.1 Structure of the TEI2.DTD File

Each parameter entity associated with a tag set controls several marked sections in the main DTD file *tei2.dtd*. If the entity has been declared in the DTD subset with the text `INCLUDE`, then the marked sections it controls will be parsed; otherwise, they will be ignored. The marked sections controlled by each entity:

1. declare and refer to the entity file for the tag set, which defines its global attributes and element classes
2. declare and refer to the DTD file for the tag set, which defines its elements and their attributes
3. declare the parameter entity *component*) in a form suitable for texts using that base

The *tei2.dtd* file has the following structure:

```
<!-- 3.6.1: File tei2.dtd: Main document type declaration     -->
<!-- file                                                     -->
<!-- Text Encoding Initiative: Guidelines for Electronic      -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.    -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy    -->
<!-- in any form is granted, provided this notice is          -->
<!-- included in all copies.                                  -->

<!-- These materials may not be altered; modifications to     -->
<!-- these DTDs should be performed as specified in the       -->
<!-- Guidelines in chapter "Modifying the TEI DTD."           -->

<!-- These materials subject to revision. Current versions    -->
<!-- are available from the Text Encoding Initiative.         -->


<!-- This file first defines some useful entities, then       -->
<!-- defines the element TEI.2 and includes files with the    -->
<!-- various specialized parts of the document type           -->
<!-- declaration. It also declares the top-level TEI.2 and    -->
<!-- TEI.2.corpus elements.                                   -->


<!-- I.  Preliminaries.                                       -->

<!-- Embed any local modifications to TEI entities.           -->

<!-- ... declarations from section 3.6.2                      -->
<!--     (Local modifications to parameter entities)          -->
<!--     go here ...                                          -->

<!-- Embed entities for TEI generic identifiers.              -->

<!ENTITY % TEI.elementNames system 'teigis2.ent'               >
%TEI.elementNames;

<!-- Define entities for TEI keywords.                        -->

<!-- This includes defining the default for each base and     -->
<!-- additional tag set as 'IGNORE'.                          -->

<!-- ... declarations from section 3.8.3                      -->
<!--     (TEI Keywords)                                       -->
<!--     go here ...                                          -->
```

```
<!-- II. Define element classes for content models, shared    -->
<!-- attributes for element classes, and global attributes.   -->
<!-- (This all happens within the file TEIclas2.ent.)         -->

<!ENTITY % TEI.elementClasses system 'teiclas2.ent'            >
%TEI.elementClasses;

<!-- III. Define the top-level TEI elements: one for          -->
<!-- individual texts, one for composites with a collective   -->
<!-- header.                                                  -->


<!-- A TEI document is a text preceded by a TEI header.       -->

<!ELEMENT TEI.2         - O  (teiHeader, text)                 >
<!ATTLIST TEI.2              %a.global;                        >

<!-- A TEI corpus is a series of TEI.2 documents, preceded    -->
<!-- by a corpus-level TEI header.                            -->

<!ELEMENT teiCorpus.2   - O  (teiHeader, TEI.2+)               >
<!ATTLIST teiCorpus.2        %a.global;                        >

<!-- IV. Embed the actual tag sets. First embed any local     -->
<!-- modifications and extensions. Then embed the core tag    -->
<!-- sets, the (single) base tag set, and the (optional)      -->
<!-- additional tag sets specified by the user.               -->

<!-- ... declarations from section 3.6.2                      -->
<!--     (Embed local element declarations, etc.)             -->
<!--     go here ...                                          -->
<!-- ... declarations from section 3.6.3                      -->
<!--     (Embed the core tag sets)                            -->
<!--     go here ...                                          -->
<!-- ... declarations from section 3.6.4                      -->
<!--     (Embed base tag set)                                 -->
<!--     go here ...                                          -->
<!-- ... declarations from section 3.6.5                      -->
<!--     (Embed additional tag sets)                          -->
<!--     go here ...                                          -->
```

A TEI-conformant document *must* use the *tei2.dtd* file, or one derived from it in the manner described in chapter 29 ('Modifying the TEI DTD') on p. 619. It must also specify which base and which additional tag sets are to be invoked, using the mechanisms described in section 3.3 ('Invocation of the TEI DTD') on p. 39.

3.6.2 Embedding Local Modifications

As noted above in section 3.2.4 ('User-Defined Tag Sets') on p. 39, local modifications to the DTD are most conveniently grouped into two files, one containing modifications to the TEI parameter entities, and the other new or changed declarations of elements and their attributes. These files should be associated with the parameter entities *TEI.extensions.ent* and *TEI.extensions.dtd* by declarations included in the document's DTD subset.

For example, if the relevant files are called *project.ent* and *project.dtd*, then declarations like the following would be appropriate:

```
<!ENTITY % TEI.extensions.ent system 'project.ent' >
<!ENTITY % TEI.extensions.dtd system 'project.dtd' >
```

When an SGML entity is declared more than once, the first declaration is binding and the others are ignored. The local modifications to parameter entities should therefore be

handled before the standard parameter entities themselves are declared in *tei2.dtd*. The entity *TEI.extensions.ent* is referred to before any TEI declarations are handled, to allow the user's declarations to take priority. If the user does not provide a *TEI.extensions.ent* entity, the entity will be expanded to the empty string.

For example the encoder might wish to add two phrase-level elements <it> and <bd>, perhaps as synonyms for <hi rend='italics'> and <hi rend='bold'>. As described in chapter 29 ('Modifying the TEI DTD') on p. 619, this involves two distinct steps: one to define the new elements, and the other to ensure that they are placed into the TEI document structure at the right place. We deal with the second first, by specifying the element class to which the new elements should be attached. To do this, the standard parameter entity *x.phrase* should be modified to include the two new generic identifiers. The file containing local declarations of the standard parameter entities will thus contain a declaration of the following form:

```
<!ENTITY % x.phrase 'it | bd |' >
```

The relevant fragment of the DTD is this:

```
<!-- 3.6.2:  Local modifications to parameter entities        -->
<!-- Embed local modifications to TEI parameter entities.     -->
<!-- Declare entity as empty string first, in case user has   -->
<!-- no mods and has not declared it.                         -->

<!ENTITY % TEI.extensions.ent ''                               >
%TEI.extensions.ent;
<!-- This fragment is used in sec. 3.6.1                      -->
```

The second type of modification needed is most conveniently performed after all the standard TEI parameter entities have been declared; this allows the element declarations provided by the user to make use of the parameter entities which define standard TEI content models and attribute definitions. To facilitate this, the parameter entity *TEI.extensions.dtd* is used to embed local element declarations *before* any of the TEI tag sets are embedded by the file *tei2.dtd*, but *after* all the TEI element classes and other parameter entities have been declared.

The task of declaring the non-standard <it> and <bd> elements is thus simplified: they can, for example, use the same parameter entities as the <hi> element. A suitable local DTD-modifications file might look like the following (note that the standard parameter-entity reference for phrase sequence is used):

```
<!ELEMENT it           - -  (%phrase.seq)                      >
<!ATTLIST it
          id               ID             #IMPLIED
          lang             IDREF          %INHERITED
          n                CDATA          #IMPLIED
          rend             CDATA          #FIXED 'italics'    >
<!ELEMENT bd           - -  (%phrase.seq)                      >
<!ATTLIST bd
          id               ID             #IMPLIED
          lang             IDREF          %INHERITED
          n                CDATA          #IMPLIED
          rend             CDATA          #FIXED 'boldface'   >
```

For further examples of local modifications to both parameter entities and element declarations, see chapter 29 ('Modifying the TEI DTD') on p. 619.

The relevant fragment of the DTD is this:

```
<!-- 3.6.2:  Embed local element declarations, etc.           -->
<!-- Embedding local modifications here allows user           -->
<!-- modifications to use all the standard TEI element        -->
<!-- classes and parameter entities.                          -->


<!ENTITY % TEI.extensions.dtd ''                               >
%TEI.extensions.dtd;
<!-- This fragment is used in sec. 3.6.1                      -->
```

3.6.3 Embedding the Core Tag Sets

The core tag sets are embedded by the file *tei2.dtd* using the parameter entities *TEI.header* and *TEI.core*. The relevant fragment of the DTD is this:

```
<!-- 3.6.3:  Embed the core tag sets                          -->
<!-- These occur in all documents and are therefore defined   -->
<!-- unconditionally.                                         -->

<!ENTITY % TEI.header.dtd system 'teihdr2.dtd'                 >
%TEI.header.dtd;
<!ENTITY % TEI.core.dtd system 'teicore2.dtd'                  >
%TEI.core.dtd;
<!-- This fragment is used in sec. 3.6.1                      -->
```

The default text structure tags, which are also documented as part of the core, are embedded by the base tag set, unless the base defines its own text structure tags; see the chapters on the individual bases.

3.6.4 Embedding the Base Tag Set

The *tei2.dtd* file embeds the appropriate files for the base tag set previously selected by means of the parameter entities described in section 3.2 (‘Core, Base, and Additional Tag Sets’) on p. 36. A parameter entity for the file containing the relevant DTD fragment is declared and referred to inside a conditional marked section controlled by the appropriate parameter entity. The relevant fragment of *tei2.dtd* is this:

```
<!-- 3.6.4:  Embed base tag set                               -->
<!-- A different base will be embedded, depending on which    -->
<!-- parameter entity has been declared by the user with the  -->
<!-- value 'INCLUDE'.                                         -->


<![ %TEI.prose [
<!ENTITY % TEI.prose.dtd system 'teipros2.dtd'                 >
%TEI.prose.dtd;
]]&nil;>


<![ %TEI.verse [
<!ENTITY % TEI.verse.dtd system 'teivers2.dtd'                 >
%TEI.verse.dtd;
]]&nil;>


<![ %TEI.drama [
<!ENTITY % TEI.drama.dtd system 'teidram2.dtd'                 >
%TEI.drama.dtd;
]]&nil;>


<![ %TEI.spoken [
<!ENTITY % TEI.spoken.dtd system 'teispok2.dtd'                >
%TEI.spoken.dtd;
]]&nil;>


<![ %TEI.dictionaries [
<!ENTITY % TEI.dictionaries.dtd system 'teidict2.dtd'          >
%TEI.dictionaries.dtd;
]]&nil;>
```

```
<![ %TEI.terminology [
<!ENTITY % TEI.terminology.dtd system 'teiterm2.dtd'            >
%TEI.terminology.dtd;
]]&nil;>


<![ %TEI.general [
<!ENTITY % TEI.general.dtd system 'teigen2.dtd'                 >
%TEI.general.dtd;
]]&nil;>


<![ %TEI.mixed [
<!ENTITY % TEI.mixed.dtd system 'teimix2.dtd'                   >
%TEI.mixed.dtd;
]]&nil;>


<!-- This fragment is used in sec. 3.6.1                       -->
```

3.6.5 Embedding the Additional Tag Sets

The *tei2.dtd* file embeds the appropriate files for any additional base tag set previously enabled by means of the parameter entities described in section 3.2 ('Core, Base, and Additional Tag Sets') on p. 36. A parameter entity for the file containing the relevant DTD fragment is declared and referred to, inside a conditional marked section controlled by the appropriate parameter entity. The relevant fragment of *tei2.dtd* is this:

```
<!-- 3.6.5:  Embed additional tag sets                         -->
<!-- These entities are declared and embedded only when the   -->
<!-- user has overridden the default declaration of IGNORE    -->
<!-- for a specific additional tag set.                       -->

<![ %TEI.linking [
<!ENTITY % TEI.linking.dtd system 'teilink2.dtd'                >
%TEI.linking.dtd;
]]&nil;>


<![ %TEI.analysis [
<!ENTITY % TEI.analysis.dtd system 'teiana2.dtd'                >
%TEI.analysis.dtd;
]]&nil;>


<![ %TEI.fs [
<!ENTITY % TEI.fs.dtd system 'teifs2.dtd'                       >
%TEI.fs.dtd;
]]&nil;>


<![ %TEI.certainty [
<!ENTITY % TEI.certainty.dtd system 'teicert2.dtd'              >
%TEI.certainty.dtd;
]]&nil;>


<![ %TEI.transcr [
<!ENTITY % TEI.transcr.dtd system 'teitran2.dtd'                >
%TEI.transcr.dtd;
```

```
]]&nil;>


<![ %TEI.textcrit [
<!ENTITY % TEI.textcrit.dtd system 'teitc2.dtd'                >
%TEI.textcrit.dtd;
]]&nil;>


<![ %TEI.names.dates [
<!ENTITY % TEI.names.dates.dtd system 'teind2.dtd'             >
%TEI.names.dates.dtd;
]]&nil;>


<![ %TEI.nets [
<!ENTITY % TEI.nets.dtd system 'teinet2.dtd'                   >
%TEI.nets.dtd;
]]&nil;>


<![ %TEI.figures [
<!ENTITY % TEI.figures.dtd system 'teifig2.dtd'                >
%TEI.figures.dtd;
]]&nil;>


<![ %TEI.corpus [
<!ENTITY % TEI.corpus.dtd system 'teicorp2.dtd'                >
%TEI.corpus.dtd;
]]&nil;>


<!-- This fragment is used in sec. 3.6.1                     -->
```

3.7 Element Classes

The TEI DTD contains over four hundred element types. To aid comprehension, modularity and modification, the majority of these elements are formally classified in some way. This section describes the various *element classes* recognized in the TEI DTD. Element classes are used to express two distinct kinds of commonality among elements. The elements of a class may share some set of SGML attributes, or they may appear in the same locations in the content models of the TEI DTDs, or both. A class is known as an *a-class* if its members share attributes, and as an *m-class* if its members appear at the same locations in the content models of other TEI elements. An element is said to *inherit* attributes, or the ability to appear at a given point in a document, from any classes of which it is a member. Classes may have subclasses and superclasses, and the characteristics of a superclass are inherited by all members of its subclasses.

Both types of element classes are represented in the TEI DTDs by parameter entities. For other uses of parameter entities in the TEI DTDs, see section 3.8 ('Other Parameter Entities in TEI DTDs') on p. 65.

This section describes the major element classes of each type together with the formal declarations for their parameter entities, which are contained in the file *teiclas2.ent*. All element classes are documented in the alphabetical reference section in Part VII.

3.7.1 Classes Which Share Attributes

An a-class groups together elements which share some set of common attributes. For example, the members of the class *names* are all elements which contain proper nouns: e.g. <name>, <placeName>, or <persName>. All of these elements use the same attributes (key and reg) to record information about the referent or the regularized form of the proper nouns. Similarly, the members of the *pointer* class share a set of attributes useful for managing cross-reference links and other pointers.[6]

The attributes shared by the members of an a-class are defined in a parameter entity; member elements inherit the attributes by referring to the parameter entity within their attribute-list declaration (examples below). This practice helps ensure that if the attribute definitions for the class change, all members of the class will automatically inherit the new definitions. Parameter entities used for this purpose form their names by taking the name of the class they define and prefixing the string 'a.'; we refer to these entities as *a-dot entities.*

For example, the declaration for the *names* class includes attribute definitions for its two attributes reg and key:

```
<!-- 3.7.1:  Sample class declaration                          -->
<!ENTITY % a.names '
          key                 CDATA               #IMPLIED
          reg                 CDATA               #IMPLIED'     >
```

Members of the class typically inherit these definitions by referring to *a.names*:

```
<!-- 3.7.1:  Sample element declarations                       -->
<!-- The members of a class refer to its a-dot entity in      -->
<!-- their attribute-list declaration, as shown here.         -->

<!ELEMENT name           - -  (%phrase.seq;)                    >
<!ATTLIST name                %a.global;
                              %a.names;
          type                CDATA               #IMPLIED      >
```

Subclasses of a-classes inherit the attributes of their superclass similarly, by referring to the a-dot entity of the superclass in defining their own a-dot entity. For example, the class *xPointer* is a subclass of the class *pointer*, as shown implicitly by the declaration of its a-dot entity:

```
<!-- 3.7.1: Sample subclass declaration, with inheritance     -->
<!-- from superclass                                          -->
<!ENTITY % a.xPointer '      %a.pointer;
          doc                 ENTITY              #IMPLIED
          from                %extPtr;            "ROOT"
          to                  %extPtr;            "DITTO"'      >
```

(For an explanation of the parameter entity *extptr* used in the above example, see section 3.8.3 ('Parameter Entities for TEI Keywords') on p. 67.)

The a-classes declared in the core tag sets of these Guidelines are:

declaring elements which have a decls attribute for specifying which declarations in the header apply to the element, as described in section 23.3 ('Associating Contextual Information with a Text') on p. 550

declarable header elements containing declarations, which can be pointed at by the decls attribute, as described in section 23.3 ('Associating Contextual Information with a Text') on p. 550

divn structural elements which behave in the same way as divisions, as described in section 7.1 ('Divisions of the Body') on p. 185

enjamb elements which carry the enjamb attribute for indicating metrical enjambement

interpret elements which contain overtly interpretive or extra-textual analysis or commentary on a text or some portion of it.

metrical elements which carry metrical information (metrical pattern, realization of the pattern, rhyme)

[6] Because the details of their pointing mechanism differ, the members of the *pointer* class do not, however, share their pointing attributes.

names elements which contain proper nouns and share attributes for identifying their referents and regularizing their spelling (section 6.4.1 ('Referring Strings') on p. 132)
personPart elements which contain personal names or parts of them
placePart elements which contain place names or parts of them
pointer elements which point from one location in the document to another (section 6.6 ('Simple Links and Cross References') on p. 147)
seg elements for the systematic or arbitrary segmentation of the text
temporalExpr elements which contain temporal expressions
timed elements (in the base tag set for spoken texts) which have a duration in time expressible with the attributes, as described in section 11.2.5 ('Temporal Information') on p. 257
xPointer elements which point from one location in the document to other locations within or outside the current document (section 14.2 ('Extended Pointers') on p. 340)

All elements are considered members of the class *global* and thus include a reference to *a.global;* in their attribute definition list declaration. Some tag sets add specialized attributes to the set of global attributes; these additions are declared in the "ent" file of each tag set, using the following entity names. If the tag set does not define new global attributes, no entity of this type is declared.

a.analysis additional global attributes for the analysis tag set
a.linking additional global attributes for the linking tag set
a.terminology additional global attributes for the terminology base

These entities are included in the *teiclas2.ent* file indirectly, when the entity-declaration files of each tag set are embedded, as shown below in section 3.7.6 ('Elements Marked for Text Type') on p. 59. For purposes of documentation, these attributes are treated as if inherited by the class *global* from superclasses called *terminology*, etc., and are documented under the class name.

One further complication to the inheritance mechanism should be mentioned here. In rare cases, a member of an a-class may override the definition of an inherited attribute. For example, the element <anchor> inherits the global id attribute from the class *global* — as does every other element. On <anchor> elements, however, id is not optional but required. The declaration of <anchor> therefore does not refer to the class *global*, but instead defines all the inherited attributes explicitly, using its own declaration for id and the default inherited declarations for the other global attributes:

```
<!-- 3.7.1: Sample of local declaration overriding inherited  -->
<!-- attributes                                                -->
<!-- ANCHOR overrides the default ID, but uses the default    -->
<!-- declarations for the other global attributes.            -->

<!ELEMENT anchor        - O  EMPTY                              >
<!ATTLIST anchor
          n                  CDATA               #IMPLIED
          lang               IDREF               %INHERITED
          rend               CDATA               #IMPLIED
                             %a.seg;
          id                 ID                  #REQUIRED      >
```

Because this declaration does not use the parameter entity *a.global*, clearly any change in the definition for that entity will not be reflected in this declaration. Consequently, any changes made to the global attributes as such will not be inherited by the <anchor> element. Instead, such changes must be replicated manually. Care must thus be taken in modifying attribute definitions for a-classes if any members of the class override the inherited definitions, to ensure that all members of the class really do get the modified definitions.

```
<!-- 3.7.1:  Attribute classes                                 -->
<!ENTITY % a.declaring '
          decls              IDREFS              #IMPLIED'      >
<!ENTITY % a.declarable '
          default            (YES | NO)          NO'            >
<!ENTITY % a.divn '
          type               CDATA               #CURRENT
          org                (composite | uniform)
```

```
                                                uniform
          sample              (initial | medial | final |
                              unknown | complete) complete
          part                (Y | N | I | M | F) N'             >
<!ENTITY % a.enjamb '
          enjamb              CDATA               #IMPLIED'      >
<!ENTITY % a.interpret '
          resp                CDATA               %INHERITED
          type                CDATA               %INHERITED
          inst                IDREFS              #IMPLIED'      >
<!ENTITY % a.metrical '
          met                 CDATA               %INHERITED
          real                CDATA               #IMPLIED
          rhyme               CDATA               #IMPLIED'      >
<!ENTITY % a.names '
          key                 CDATA               #IMPLIED
          reg                 CDATA               #IMPLIED'      >
<!ENTITY % a.personPart '     %a.names;
          type                CDATA               #IMPLIED
          full                (yes | abb | init)  yes
          sort                NUMBER              #IMPLIED'      >
<!ENTITY % a.placePart '      %a.names;
          type                CDATA               #IMPLIED
          full                (yes | abb | init)  yes'           >
<!ENTITY % a.pointer '
          type                CDATA               #IMPLIED
          resp                CDATA               #IMPLIED
          crdate              CDATA               #IMPLIED
          targType            NAMES               #IMPLIED
          targOrder           (Y | N | U)         U
          evaluate            (all | one | none)  #IMPLIED'      >
<!ENTITY % a.seg '
          type                CDATA               #IMPLIED
          function            CDATA               #IMPLIED'      >
<!ENTITY % a.temporalExpr '
          value               CDATA               #IMPLIED
          type                CDATA               #IMPLIED
          reg                 CDATA               #IMPLIED
          full                (yes | abb | init)  yes'           >
<!ENTITY % a.timed '
          start               IDREF               #IMPLIED
          end                 IDREF               #IMPLIED
          dur                 CDATA               #IMPLIED'      >
<!ENTITY % a.xPointer '       %a.pointer;
          doc                 ENTITY              #IMPLIED
          from                %extPtr;            "ROOT"
          to                  %extPtr;            "DITTO"'       >
<!-- This fragment is used in sec. 3.7.3                         -->
```

3.7.2 Classes Used in Content Models

When the members of a class are structurally similar and can appear at the same kinds of structural locations in the document, they are grouped together into an m-class (or "model-class"). M-classes are implemented by defining a parameter entity for use in the formal declaration of element content models. The parameter entity takes the name of the class it defines, and prefixes the string 'm.', which can be interpreted as 'model' or as 'members'. The replacement text of the entity is a list of the members of the class, separated by '|', the SGML symbol for alternation.

For each class an additional entity is defined, which also takes the name of the class, this time prefixed by the string 'x.' (for extension); the default value of these *x-dot entities* is always the empty

string. A reference to the corresponding x-dot entity is always included within the replacement string for each m-dot entity. This enables an encoder to add new members to a class simply by declaring a new value for an x-dot entity.

For example, the class *bibl* has the three members <bibl>, <bibl.full>, and <bibl.struct>. Its content-model entity is defined thus:

```
<!ENTITY % x.bibl '' >
<!ENTITY % m.bibl '%x.bibl bibl | bibl.full | bibl.struct' >
```

With the default value of the x-dot entity, this is the same as defining *m.bibl* with the replacement text `bibl | bibl.full | bibl.struct`. If an encoder wishes to add a new bibliographic element called <myBib>, it can be added to the *bibl* class by redefining the x-dot entity thus:

```
<!ENTITY % x.bibl 'myBib |' >
```

This changes the replacement text of *m.bibl* from its default value to `myBib | bibl | bibl.full | bibl.struct`. If more than one element is to be added to a class, the x-dot entity for the class should be redefined as a list of the new generic identifiers, each one *(including the last)* followed by a vertical bar. The same effect could be achieved simply by redefining the whole of the new *m.bibl* entity directly, but the x-dot method requires no repetition of the already existing members of the class and thus minimizes the chance of error.

Like a-classes, m-classes may have subclasses or superclasses. Just as elements inherit from a class the ability to appear in certain locations of a document (wherever the class can appear), so all members of a subclass inherit the ability to appear wherever any superclass can appear. Superclasses transmit their location characteristics to their subclasses by referring, in declaring their m-dot entity, to the m-dot entities of the subclasses.

For example, the class *phrase* includes the classes *data*, *edit*, *hqphrase*, *loc*, and *seg* as members, as can be seen in the declaration for its m-dot entity:

```
<!-- 3.7.2: Sample class declaration with inclusion of        -->
<!-- subclasses                                               -->
<!ENTITY % x.phrase ''                                         >
<!ENTITY % m.phrase '%x.phrase %m.data; | %m.edit; |
           %m.formPointers; | %m.hqphrase; | %m.loc; |
           %m.phrase.verse; | %m.seg; | %m.sgmlKeywords; |
           formula | handShift'                                >
```

When the entity *m.phrase* is referred to in content models, all members of all subclasses are included in the model.

3.7.3 The TEICLAS2.ENT File

The most important element classes used in TEI content models are declared in the DTD file *teiclas2.ent*, which is the default replacement text for the entity *TEI.elementClasses* and is embedded by the *tei2.dtd* file. These element classes are described, and their declarations reproduced, in the following sections.

The class system is structured around the following threefold division of elements:

chunks elements such as paragraphs and other paragraph-level elements, which can appear directly within texts or within text subdivisions (i.e. elements), but not within other chunks

phrase-level elements elements such as highlighted phrases, book titles, or editorial corrections which can occur only within chunks (paragraphs or paragraph-level elements), but not between them (and thus cannot appear directly within a) [7]

inter-level elements elements such as lists, notes, quotations, etc. which can appear either between chunks (as children of a) or within them

Together the two sets of *chunks* and *inter-level elements* make up the set of:

[7] Note that in this context, *phrase* means any string of characters, and can apply to individual words, parts of words, and groups of words indifferently; it does not refer only to linguistically motivated phrasal units. This may cause confusion for readers accustomed to applying the word in a more restrictive sense.

text components elements which can appear directly within texts or text divisions; also called simply *components* or "component-level elements"

In general, the body of any text comprises a series of components, optionally grouped into elements.

Some elements belong to none of these classes; these include high-level structural elements like <tei.2> and <group> as well as some specialized elements which appear only within particular structures (like <analytic>, <monographic>, and <series>). The majority of elements found in normal running text, however, are assigned by the TEI DTDs to one or the other of these classes.

Some *component* elements (e.g. <p> or <note>) are common to all base tag sets, while others are unique to individual tag sets. This distinction is reflected in the parameter entity declarations, as shown below.

The *teiclas2.ent* file has the following overall structure:

```
<!-- 3.7.3:  Element classes for TEI DTDs                    -->
<!-- Text Encoding Initiative: Guidelines for Electronic     -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.   -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy   -->
<!-- in any form is granted, provided this notice is         -->
<!-- included in all copies.                                 -->

<!-- These materials may not be altered; modifications to    -->
<!-- these DTDs should be performed as specified in the      -->
<!-- Guidelines in chapter "Modifying the TEI DTD."          -->

<!-- These materials subject to revision. Current versions   -->
<!-- are available from the Text Encoding Initiative.        -->


<!-- First, we declare the 'low-level' core classes: these   -->
<!-- are classes of semantically and structurally similar    -->
<!-- elements declared as part of the core tag set, e.g. the -->
<!-- classes 'data' or 'edit'.                               -->


<!-- ... declarations from section 3.7.4                     -->
<!--     (Low-level classes)                                 -->
<!--     go here ...                                         -->
<!-- ... declarations from section 3.7.9                     -->
<!--     (Misc. Element Class Models)                        -->
<!--     go here ...                                         -->

<!-- Next, we declare the 'high-level' classes: these group  -->
<!-- together all phrase-level elements, all inter-level     -->
<!-- elements, and all chunk-level elements in the core, and -->
<!-- identify the 'common' component elements (chunks and    -->
<!-- inter-level elements), as opposed to the                -->
<!-- tagset-specific components.                             -->


<!-- ... declarations from section 3.7.5                     -->
<!--     (Common high-level classes)                         -->
<!--     go here ...                                         -->

<!-- Next, we embed the portions of each base and additional -->
<!-- tag set which declare relevant parameter entities. Only -->
<!-- those files are embedded which have been selected by    -->
<!-- the user in the DTD subset. These files will declare    -->
<!-- parameter entities for their component-level elements,  -->
```

```
<!-- as well as for any global attributes they define.       -->


<!-- ... declarations from section 3.7.6                     -->
<!--     (Embedding tag-set-specific entity definitions)     -->
<!--     go here ...                                         -->

<!-- We can now declare the standard content models; one of  -->
<!-- these varies with the base selected.                    -->

<!-- ... declarations from section 3.7.7                     -->
<!--     (Standard Content Models)                           -->
<!--     go here ...                                         -->

<!-- Finally, we declare the attribute classes, including    -->
<!-- the global attributes.                                  -->

<!-- ... declarations from section 3.7.1                     -->
<!--     (Attribute classes)                                 -->
<!--     go here ...                                         -->
<!-- ... declarations from section 3.5                       -->
<!--     (Global attributes)                                 -->
<!--     go here ...                                         -->
```

3.7.4 Low-Level Element Classes

The following low-level classes group together sets of semantically or structurally similar elements. These classes may include both elements in the core and elements declared in particular tag sets; a reference is given at least to the relevant section on the core tags.

The following are phrase-level element classes:

hqphrase elements for highlighted phrases or material marked by quotation marks, including those defined in section 6.3 ('Highlighting and Quotation') on p. 123

data elements for recording information about the referents of a text, including those defined in section 6.4 ('Names, Numbers, Dates, Abbreviations, and Addresses') on p. 132

date elements for recording dates, including those defined in section 6.4.4 ('Dates and times') on p. 137

edit elements for recording simple editorial interventions in a text, including those defined in section 6.5 ('Simple Editorial Changes') on p. 140

loc elements for recording location information in a text, including those defined in section 6.9 ('Reference Systems') on p. 155

seg elements for marking arbitrary segments at the level of individual characters or phrases, including those documented in section 14.3 ('Segments and Anchors') on p. 355 and 15.1 ('Linguistic Segment Categories') on p. 382

sgmlKeywords elements for marking generic identifiers, attribute names, SGML tags, and sample attribute values, when they occur in the text (used in tag set documentation, for which see chapter 27 ('Tag Set Documentation') on p. 601)

versePhrases phrase-level elements specific to verse, documented in section 9.3 ('Components of the Verse Line') on p. 218

formPointers elements for referring, within a dictionary entry, to the orthographic form or pronunciation of the headword, documented in section 12.4 ('Headword and Pronunciation References') on p. 297

The following are inter-level element classes:

hqinter elements for highlighted phrases or material marked by quotation marks, including those defined in section 6.3 ('Highlighting and Quotation') on p. 123

bibl elements for bibliographic citations; see section 6.10 ('Bibliographic Citations and References') on p. 162

lists elements for lists; see section 6.7 ('Lists') on p. 149

notes general-purpose annotation elements; see section 6.8 ('Notes, Annotation, and Indexing') on p. 152

stageDirection elements for specialized stage-direction elements documented in section 10.2.3 ('Stage Directions') on p. 238

The following classes of elements may appear anywhere within the <text> element:

metadata elements which convey non-textual information about the text (meta-information, as it were)

globincl elements which may appear anywhere within the <text> element (because the class is used in an inclusion exception on that element)

The entity declarations for these classes are these:

```
<!-- 3.7.4:  Low-level classes                                  -->
<!-- Most of these elements are in the core tag set, but       -->
<!-- some may be from other tag sets.                          -->


<!-- Phrase-level classes                                       -->

<!ENTITY % x.hqphrase ''                                        >
<!ENTITY % m.hqphrase '%x.hqphrase distinct | emph | foreign |
           gloss | hi | mentioned | soCalled | term | title'    >
<!ENTITY % x.date ''                                            >
<!ENTITY % m.date '%x.date date | dateRange | dateStruct'       >
<!ENTITY % x.data ''                                            >
<!ENTITY % m.data '%x.data abbr | address | date | dateRange |
           dateStruct | expan | lang | measure | name | num |
           rs | time | timeRange | timeStruct'                 >
<!ENTITY % x.edit ''                                            >
<!ENTITY % m.edit '%x.edit add | corr | del | gap | orig | reg
           | sic'                                               >
<!ENTITY % x.loc ''                                             >
<!ENTITY % m.loc '%x.loc link | ptr | ref | xptr | xref'        >
<!ENTITY % x.seg ''                                             >
<!ENTITY % m.seg '%x.seg anchor | c | cl | m | phr | s | seg |
           w'                                                   >
<!ENTITY % x.sgmlKeywords ''                                    >
<!ENTITY % m.sgmlKeywords '%x.sgmlKeywords att | gi | tag |
           val'                                                 >
<!ENTITY % x.phrase.verse ''                                    >
<!ENTITY % m.phrase.verse '%x.phrase.verse caesura'             >
<!ENTITY % x.formPointers ''                                    >
<!ENTITY % m.formPointers '%x.formPointers oRef | oVar | pRef |
           pVar'                                                >
<!ENTITY % x.metadata ''                                        >
<!ENTITY % m.metadata '%x.metadata certainty | interp |
           interpGrp | respons | span | spanGrp'               >
<!ENTITY % x.globincl ''                                        >
<!ENTITY % m.globincl '%x.globincl %m.metadata;'                >

<!-- Inter-level classes                                       -->

<!ENTITY % x.hqinter ''                                         >
<!ENTITY % m.hqinter '%x.hqinter cit | q | quote'               >
<!ENTITY % x.bibl ''                                            >
<!ENTITY % m.bibl '%x.bibl bibl | biblFull | biblStruct'        >
<!ENTITY % x.lists ''                                           >
<!ENTITY % m.lists '%x.lists label | list | listBibl'           >
<!ENTITY % x.notes ''                                           >
<!ENTITY % m.notes '%x.notes note'                              >
```

```
<!ENTITY % x.stageDirection ''                                    >
<!ENTITY % m.stageDirection '%x.stageDirection camera | caption
          | move | sound | tech | view'                           >
<!-- This fragment is used in sec. 3.7.3                        -->
```

3.7.5 High-Level Element Classes

The following element classes are used to implement the threefold structural distinction among phrases, chunks, and intermediate elements discussed above in section 3.7.3 ('The TEICLAS2.ENT File') on p. 55. In this terminology, *chunk*s (or *chunk elements* are elements which can occur only in chunk-level sequences (e.g. between but not within paragraphs); *inter-level elements* can occur either within chunks (at phrase-level) or between chunks (e.g. at paragraph-level), and *phrase-level elements* can occur only at phrase level, within chunks (e.g. within but not between paragraphs).

The element class *common* includes all component-level (chunk- and inter-level) elements common to more than one base. It is used in implementing the combined bases described in section 3.4 ('Combining TEI Base Tag Sets') on p. 40.

The relevant portion of the DTD looks like this:

```
<!-- 3.7.5:  Common high-level classes                          -->
<!-- These are the three fundamental element classes.           -->

<!ENTITY % x.phrase ''                                            >
<!ENTITY % m.phrase '%x.phrase %m.data; | %m.edit; |
           %m.formPointers; | %m.hqphrase; | %m.loc; |
           %m.phrase.verse; | %m.seg; | %m.sgmlKeywords; |
           formula | handShift'                                   >
<!ENTITY % x.inter ''                                             >
<!ENTITY % m.inter '%x.inter %m.bibl; | %m.hqinter; | %m.lists;
           | %m.notes; | %m.stageDirection; | castList | figure
           | stage | table | text'                                >
<!ENTITY % x.chunk ''                                             >
<!ENTITY % m.chunk '%x.chunk eTree | graph | l | lg | p | sp |
           tree'                                                  >

<!-- This class isolates all the common component-level         -->
<!-- elements.                                                  -->

<!ENTITY % x.common ''                                            >
<!ENTITY % m.common '%x.common %m.bibl; | %m.chunk; |
           %m.hqinter; | %m.lists; | %m.notes; | stage'           >
<!-- This fragment is used in sec. 3.7.3                        -->
```

3.7.6 Elements Marked for Text Type

The following element classes are used to group together component-level elements which are allowed only in texts of a particular type (i.e. texts using a specific base).

comp.verse elements unique to verse
comp.drama elements unique to drama
comp.spoken elements unique to spoken texts
comp.dictionaries elements unique to dictionaries
comp.terminology elements unique to terminological data

Declarations for these base-specific element classes are included in the *entity file* of each base, which is in turn embedded by the *teiclas2.dtd* file in the DTD fragment shown below. If the tag set defines additions to the set of global attributes, or declares a class of component-level elements unique to the tag set, then it has an entity file which is embedded here; otherwise not.

```
<!-- 3.7.6:  Embedding tag-set-specific entity definitions     -->
```

```
<![ %TEI.verse [
<!ENTITY % TEI.verse.ent system 'teivers2.ent'                    >
%TEI.verse.ent;
]]&nil;>


<![ %TEI.drama [
<!ENTITY % TEI.drama.ent system 'teidram2.ent'                    >
%TEI.drama.ent;
]]&nil;>


<![ %TEI.spoken [
<!ENTITY % TEI.spoken.ent system 'teispok2.ent'                   >
%TEI.spoken.ent;
]]&nil;>


<![ %TEI.dictionaries [
<!ENTITY % TEI.dictionaries.ent system 'teidict2.ent'             >
%TEI.dictionaries.ent;
]]&nil;>


<![ %TEI.terminology [
<!ENTITY % TEI.terminology.ent system 'teiterm2.ent'              >
%TEI.terminology.ent;
]]&nil;>


<![ %TEI.linking [
<!ENTITY % TEI.linking.ent system 'teilink2.ent'                  >
%TEI.linking.ent;
]]&nil;>


<![ %TEI.analysis [
<!ENTITY % TEI.analysis.ent system 'teiana2.ent'                  >
%TEI.analysis.ent;
]]&nil;>


<![ %TEI.transcr [
<!ENTITY % TEI.transcr.ent system 'teitran2.ent'                  >
%TEI.transcr.ent;
]]&nil;>


<![ %TEI.textcrit [
<!ENTITY % TEI.textcrit.ent system 'teitc2.ent'                   >
%TEI.textcrit.ent;
]]&nil;>


<![ %TEI.names.dates [
<!ENTITY % TEI.names.dates.ent system 'teind2.ent'                >
%TEI.names.dates.ent;
]]&nil;>
```

```
<![ %TEI.figures [
<!ENTITY % TEI.figures.ent system 'teifig2.ent'                >
%TEI.figures.ent;
]]&nil;>


<!-- This fragment is used in sec. 3.7.3                      -->
```

3.7.7 Standard Content Models

As far as possible, the TEI DTDs use the following set of frequently-encountered content models, to help achieve consistency among different elements.

phrase a single sequence of character data or single phrase-level element

phrase.seq sequence of character data and phrase-level elements

component a single chunk- or inter-level element

component.seq sequence of chunk- and inter-level elements; this is the usual content of a element

paraContent sequence of character data, phrase-level elements, and inter-level elements; this is the usual content of chunks (including, most prominently, paragraphs)

specialPara specialized content model, allowing *either* a sequence of chunks *or* the same content as *paraContent*; this is used for elements such as notes and list items, which can behave either as chunk-level elements, or else as containers for groups of component-level elements.

The relevant portion of the DTD looks like this:

```
<!-- 3.7.7:  Standard Content Models                          -->

<!-- Here we declare the parameter entities phrase,           -->
<!-- phrase.seq, component, component.seq, paraContent, and   -->
<!-- specialPara, for use in the content models of element    -->
<!-- declarations.                                            -->


<!-- The entities phrase and phrase.seq are the same in all   -->
<!-- bases. They may include elements specific to single tag  -->
<!-- sets; if the tag set is not selected, these elements     -->
<!-- are undefined and have no effect.                        -->


<!ENTITY % phrase '(#PCDATA | %m.phrase)'                      >
<!ENTITY % phrase.seq '(%phrase;)*'                            >

<!-- The entity component varies with the base. The versions  -->
<!-- for the combined bases are declared first (so as to      -->
<!-- take precedence over the declarations in the individual  -->
<!-- bases).                                                  -->


<!-- ... declarations from section 3.7.8                      -->
<!--     (Definition of components for combined bases)        -->
<!--     go here ...                                          -->

<![ %TEI.prose [
<!ENTITY % component '(%m.common)'                             >
<!ENTITY % TEI.singleBase 'INCLUDE'                            >
]]&nil;>


<![ %TEI.verse [
<!ENTITY % component '(%m.common | %m.comp.verse)'             >
<!ENTITY % TEI.singleBase 'INCLUDE'                            >
```

```
]]&nil;>


<![ %TEI.drama [
<!ENTITY % component '(%m.common | %m.comp.drama)'              >
<!ENTITY % TEI.singleBase 'INCLUDE'                             >
]]&nil;>


<![ %TEI.spoken [
<!ENTITY % component '(%m.common | %m.comp.spoken)'              >
<!ENTITY % TEI.singleBase 'INCLUDE'                             >
]]&nil;>


<![ %TEI.dictionaries [
<!ENTITY % component '(%m.common | %m.comp.dictionaries)'       >
<!ENTITY % TEI.singleBase 'INCLUDE'                             >
]]&nil;>


<![ %TEI.terminology [
<!ENTITY % component '(%m.common | %m.comp.terminology)'        >
<!ENTITY % TEI.singleBase 'INCLUDE'                             >
]]&nil;>


<!-- Default declaration.                                      -->

<!ENTITY % component '(%m.common)'                              >
<!ENTITY % TEI.singleBase 'INCLUDE'                             >

<!-- The entity component.seq is always a starred sequence    -->
<!-- of component elements. Its definition does not vary      -->
<!-- with the base (unless we are using the general base, in  -->
<!-- which case it has already been defined above), but the   -->
<!-- meaning of the definition does.                           -->

<!ENTITY % component.seq '(%component;)*'                       >

<!-- The following entities do not vary with the base.        -->

<!ENTITY % paraContent '(#PCDATA | %m.phrase | %m.inter)*'       >
<!ENTITY % specialPara '(((%m.chunk), (%component.seq)) |
(%paraContent))'                                                >
<!-- This fragment is used in sec. 3.7.3                       -->
```

3.7.8 Components in Mixed and General Bases

When the mixed or general base is in use, the definitions of the entities *component* and *component.seq* are rather more complex. The relevant portion of the DTD is this:

```
<!-- 3.7.8:  Definition of components for combined bases      -->

<!-- Default declarations for the 'mix.' entities used for    -->
<!-- mixed and general bases.                                  -->

<!ENTITY % mix.verse ''                                         >
<!ENTITY % mix.drama ''                                         >
<!ENTITY % mix.spoken ''                                        >
<!ENTITY % mix.dictionaries ''                                  >
```

```
<!ENTITY % mix.terminology '' >
<![ %TEI.mixed [
<!ENTITY % TEI.singleBase 'IGNORE' >
<!-- The mixed base allows components from any base; it does -->
<!-- so by defining 'component' as including both common -->
<!-- components and those specific to one tag set. -->

<!ENTITY % component '(%m.common %mix.verse %mix.drama
%mix.spoken
%mix.dictionaries %mix.terminology)' >
]]&nil;>


<![ %TEI.general [
<!-- The general base uses the same definition of component -->
<!-- as the mixed base. -->

<!ENTITY % TEI.singleBase 'IGNORE' >
<!ENTITY % component '(%m.common %mix.verse %mix.drama
%mix.spoken
%mix.dictionaries %mix.terminology)' >
<!-- But it defines a special version of component.seq, -->
<!-- which restricts each div of the text to a single base: -->
<!-- bases can shift only in embedded divs or at div -->
<!-- boundaries. This entity is constructed out of a series -->
<!-- of smaller entities, one for each tag set. If the tag -->
<!-- set is not in use, its entity will expand to the empty -->
<!-- string. -->


<![ %TEI.verse [
<!-- If the verse base is in use, ... -->

<!ENTITY % gen.verse '((%m.comp.verse), (%m.common |
%m.comp.verse)*) |' >
]]&nil;>


<![ %TEI.drama [
<!-- If the drama base is in use, ... -->

<!ENTITY % gen.drama '((%m.comp.drama), (%m.common |
%m.comp.drama)*) |' >
]]&nil;>


<![ %TEI.spoken [
<!-- If the spoken base is in use, ... -->

<!ENTITY % gen.spoken '((%m.comp.spoken), (%m.common |
%m.comp.spoken)*) |' >
]]&nil;>


<![ %TEI.dictionaries [
<!-- If the dictionary base is in use, ... -->

<!ENTITY % gen.dictionaries '((%m.comp.dictionaries),
(%m.common | %m.comp.dictionaries)*) |' >
]]&nil;>
```

```
<![ %TEI.terminology [
<!-- If the terminology base is in use, ...                    -->

<!ENTITY % gen.terminology '((%m.comp.terminology), (%m.common
| %m.comp.terminology)*) |'                                      >
]]&nil;>


<!-- Default declarations for all the entities gen.verse,      -->
<!-- etc.                                                      -->

<!ENTITY % gen.verse ''                                         >
<!ENTITY % gen.drama ''                                         >
<!ENTITY % gen.spoken ''                                        >
<!ENTITY % gen.dictionaries ''                                  >
<!ENTITY % gen.terminology ''                                   >

<!-- Now we are ready to declare component.seq and             -->
<!-- component.plus for use in general base tag set.           -->

<!ENTITY % component.seq '(%m.common)*, (%gen.verse %gen.drama
%gen.spoken %gen.dictionaries
%gen.terminology TEI...end)?'                                   >
<!ENTITY % component.plus '(%gen.verse %gen.drama %gen.spoken
%gen.dictionaries %gen.terminology
TEI...end)
|
(
(%m.common)+, (%gen.verse %gen.drama %gen.spoken
%gen.dictionaries
%gen.terminology TEI...end)?'                                   >

<!-- (End of marked section for general base.)                 -->

]]&nil;>


<!-- This fragment is used in sec. 3.7.7                       -->
```

3.7.9 Miscellaneous Content-Model Classes

The following element classes occupy specific places in content models; some are relevant only when certain tag sets are selected

agent elements which denote an individual or organization to whom or which responsibility for an action can be assigned

addrPart elements which can occur as part of an address

biblPart elements which can occur in bibliographic citations

demographic elements which record demographic characteristics of the participants in a text or language interaction (used in tag set for corpora and collections)

divbot elements which can occur as part of the closing material of a text division or body

divtop elements which can occur as part of the opening material of a text division or body

dramafront elements which can occur in the front matter of drama and other performance texts

front elements which can occur (at the level of text divisions) in front matter only

personPart elements which contain parts of a personal name

placePart elements which contain parts of a place name

refsys milestone elements used in reference systems

tpParts elements which occur within title pages

They are declared in the following DTD fragment:

```
<!-- 3.7.9:  Misc. Element Class Models                          -->
<!ENTITY % x.agent ''                                            >
<!ENTITY % m.agent '%x.agent name'                               >
<!ENTITY % x.addrPart ''                                         >
<!ENTITY % m.addrPart '%x.addrPart postBox | postCode | street' >
<!ENTITY % x.biblPart ''                                         >
<!ENTITY % m.biblPart '%x.biblPart analytic | author |
           biblScope | edition | editor | extent | idno |
           imprint | monogr | note | publisher | pubPlace |
           respStmt | series'                                    >
<!ENTITY % x.demographic ''                                      >
<!ENTITY % m.demographic '%x.demographic birth | education |
           firstLang | occupation | persName | residence |
           socecStatus'                                          >
<!ENTITY % x.divbot ''                                           >
<!ENTITY % m.divbot '%x.divbot byline | closer | epigraph |
           salute | signed | trailer'                           >
<!ENTITY % x.divtop ''                                           >
<!ENTITY % m.divtop '%x.divtop argument | byline | docAuthor |
           docDate | epigraph | head | opener | salute |
           signed'                                               >
<!ENTITY % x.dramafront ''                                       >
<!ENTITY % m.dramafront '%x.dramafront epilogue | performance |
           prologue | set'                                      >
<!ENTITY % x.front ''                                            >
<!ENTITY % m.front '%x.front %m.dramafront; | titlePage'         >
<!ENTITY % x.personPart ''                                       >
<!ENTITY % m.personPart '%x.personPart addName | forename |
           genName | nameLink | roleName | surname'             >
<!ENTITY % x.placePart ''                                        >
<!ENTITY % m.placePart '%x.placePart bloc | country | distance
           | geog | geogName | offset | placeName | region |
           settlement'                                           >
<!ENTITY % x.refsys ''                                           >
<!ENTITY % m.refsys '%x.refsys cb | lb | milestone | pb'         >
<!ENTITY % x.tpParts ''                                          >
<!ENTITY % m.tpParts '%x.tpParts byline | docAuthor | docDate |
           docEdition | docImprint | docTitle | epigraph |
           imprimatur | titlePart'                              >
<!-- This fragment is used in sec. 3.7.3                       -->
```

3.8 Other Parameter Entities in TEI DTDs

The TEI DTDs use SGML parameter entities for several purposes:

- to define sets of attributes shared by given classes of elements
- to define classes of elements which can occur at the same locations in content models
- to identify what base tag set should be used for a document
- to identify what additional tag sets should be included
- to include or exclude the declaration of each element
- to specify the name of each element

The first two applications of parameter entities are described above in section 3.7 (‘Element Classes’) on p. 51. This chapter describes the other uses of parameter entities in the TEI DTDs.

The parameter entities used to specify which base tag set and which additional tag sets are to be used in a given document are listed in section 3.2 (‘Core, Base, and Additional Tag

Sets') on p. 36. Their default definition is always "IGNORE": the encoder selects the TEI base and additional tag sets by declaring the appropriate parameter entities with the entity text "INCLUDE".

The DTD and entity files are listed in section 3.2 ('Core, Base, and Additional Tag Sets') on p. 36; if the standard TEI entities are modified to point at locally developed DTD files containing local modifications or extensions to the TEI DTDs, the use of the standard parameter entity names ensures that the modification will be obvious upon examination of the document's DTD.

The following entities are referred to by the main *tei2.dtd* file to embed portions of the TEI DTDs or locally developed extensions.

TEI.extensions.ent identifies a local file containing extensions to the TEI parameter entities; see section 3.6.2 ('Embedding Local Modifications') on p. 47

TEI.extensions.dtd identifies a local file containing extensions to the TEI tag set; see section 3.6.2 ('Embedding Local Modifications') on p. 47

TEI.elementNames identifies a file containing parameter entity declarations for names of TEI elements; see section 3.8.2 ('Parameter Entities for Element Generic Identifiers') on p. 66

TEI.keywords identifies a file containing parameter entity declarations for TEI keywords, including the default declaration ("IGNORE") of the marked-section keyword for each tag set; see section 3.8.3 ('Parameter Entities for TEI Keywords') on p. 67

TEI.elementClasses identifies a file containing definitions of parameter entities used in content models; see section 3.7.3 ('The TEICLAS2.ENT File') on p. 55

TEI.singleBase defined as `INCLUDE` (for normal bases) or `IGNORE` (for mixed and general base); used to prevent multiple definitions of the default text structure.

3.8.1 Inclusion and Exclusion of Elements

The TEI DTDs use marked sections and parameter entity references to allow users to exclude the definitions of individual elements, in order either to make the elements illegal in a document or to allow the element to be redefined, as further described in chapter 29 ('Modifying the TEI DTD') on p. 619.

Parameter entities used for this purpose have exactly the same name as the generic identifier of the element concerned. The default definition for these parameter entities is "INCLUDE" but they may be changed to "IGNORE" in order to exclude the standard element and attribute definition list declarations from the DTD.

The declarations for the element <p>, for example, are preceded by a definition for a parameter entity with the name *p* and contained within a marked section whose keyword is given as "%p;":

```
<!ENTITY % p 'INCLUDE' >
<![ %p; [
        <!-- element and attlist declaration for p here -->
]]>
```

These parameter entities are defined immediately preceding the element whose declarations they control; because their names are completely regular, they are not documented individually in the reference section of this document.

3.8.2 Parameter Entities for Element Generic Identifiers

In the TEI DTDs, elements are not referred to directly by their generic identifiers; instead, the DTDs refer to parameter entities which expand to the standard generic identifiers. This allows users to rename elements by redefining the appropriate parameter entity (as described more fully in chapter 29 ('Modifying the TEI DTD') on p. 619). Parameter entities used for this purpose are formed by taking the standard name (generic identifier) of the element and attaching the string "n." as a prefix. Thus the standard generic identifiers for paragraphs, notes, and quotations, <p>, <note>, and <q> are defined by declarations of the following form:

```
<!ENTITY % n.p    'p' >
<!ENTITY % n.note 'note' >
<!ENTITY % n.q    'q' >
```

Since by default parameter entities are case-sensitive, the specific mix of upper and lower case letters in the standard name must be preserved in the entity name.

The formal declarations of the parameter entities used for generic identifiers are contained in the file *teigis2.ent*; since their names and replacement texts are fully predictable, these parameter entities are not individually documented in the reference section of these Guidelines. The parameter entity *TEI.elementNames* is used to embed the file *teigis2.ent* in the DTD. A full set of alternate generic identifiers can be substituted for the standard set by defining *TEI.elementNames* to point at a different file.[8]

3.8.3 Parameter Entities for TEI Keywords

The TEI uses the following parameter entities to signal information which cannot be expressed using SGML keywords:

INHERITED indicates that an attribute value is inherited from the enclosing element, if not specified

ISO-date indicates that an attribute value should be a legal ISO date in the form yyyy-mm-dd (e.g. 1993-06-28).

extptr indicates that an attribute value should be a legal expression in the TEI extended-pointer notation

In addition, the parameter entities which control the selection of base and additional tag sets may be regarded as a keyword.

The parameter entity *INHERITED* is used to signal that the default value for an attribute should be inherited from an enclosing element. The definition for *INHERITED* is the string "#IMPLIED"; as for all implied defaults, the application program is responsible for deducing the default attribute value when no value is specified in the element start-tag. Since the parameter entity is resolved by the SGML parser, the application program will see no difference between attributes whose default is "%INHERITED" and those whose default is "#IMPLIED" — information about which attribute values are inherited and which are inferred in some other way must be built into the application in advance.

The parameter entity *ISO-date* is used to signal that the value for an attribute should be an ISO-standard date value; in this notation,[9] a date like "September 22, 1968" would be written "1968-09-22". The parameter entity *ISO-date* expands to "CDATA".

The keywords controlling the selection of base and additional tag sets (described in section 3.2 ('Core, Base, and Additional Tag Sets') on p. 36) all have the default value "IGNORE"; the user can override this by a local declaration, as described in section 3.3 ('Invocation of the TEI DTD') on p. 39.

The parameter entities for TEI keywords are included in file *teikey2.dtd*, which is the default replacement text for the entity *TEI.keywords* and is embedded by the file *tei2.dtd*.

The file *teikey2.dtd* has the following contents:

```
<!-- 3.8.3:  TEI Keywords                                    -->

<!-- I.  Declare TEI keywords for data types.                -->

<!-- These parameter entities are used as keywords to        -->
<!-- express rules or constraints which cannot be fully      -->
<!-- expressed in SGML; their expansions show the nearest    -->
<!-- SGML equivalent.                                        -->


<!ENTITY % INHERITED '#IMPLIED'                               >
<!ENTITY % ISO-date 'CDATA'                                   >
```

[8] It is expected that after completion of the full text of these Guidelines, the TEI will prepare alternate sets of generic identifiers in languages other than English. It should be noted, however, that in the interests of simplicity parameter entities are used only for generic identifiers; attribute names, standard attribute values, and parameter entity names are less easily modified.

[9] Defined by ISO 8601: 1988, *Data elements and interchange formats — Information interchange — Representation of dates and times* ([Geneva]: International Organization for Standardization, 1988).

```
<!ENTITY % extPtr 'CDATA'                                          >

<!-- II.  Declare keywords for tag-set selection.                 -->

<!-- Declare all bases and additional tag sets as 'IGNORE'.       -->
<!-- The user can override this default by declaring the          -->
<!-- same entity with the replacement text 'INCLUDE', in the      -->
<!-- document's DTD subset.                                       -->

<!-- Base tag sets first.                                         -->

<!ENTITY % TEI.prose 'IGNORE'                                      >
<!ENTITY % TEI.verse 'IGNORE'                                      >
<!ENTITY % TEI.drama 'IGNORE'                                      >
<!ENTITY % TEI.spoken 'IGNORE'                                     >
<!ENTITY % TEI.dictionaries 'IGNORE'                               >
<!ENTITY % TEI.terminology 'IGNORE'                                >

<!-- Now the mixed bases.                                         -->

<!ENTITY % TEI.general 'IGNORE'                                    >
<!ENTITY % TEI.mixed 'IGNORE'                                      >

<!-- Now additional tag sets.                                     -->

<!ENTITY % TEI.linking 'IGNORE'                                    >
<!ENTITY % TEI.analysis 'IGNORE'                                   >
<!ENTITY % TEI.fs 'IGNORE'                                         >
<!ENTITY % TEI.certainty 'IGNORE'                                  >
<!ENTITY % TEI.transcr 'IGNORE'                                    >
<!ENTITY % TEI.textcrit 'IGNORE'                                   >
<!ENTITY % TEI.names.dates 'IGNORE'                                >
<!ENTITY % TEI.nets 'IGNORE'                                       >
<!ENTITY % TEI.figures 'IGNORE'                                    >
<!ENTITY % TEI.figures 'IGNORE'                                    >
<!ENTITY % TEI.corpus 'IGNORE'                                     >
```

The relevant portion of the main DTD looks like this:

```
<!-- 3.8.3:  TEI Keywords                                         -->
<!-- We declare and immediately embed the TEI keywords file.      -->


<!ENTITY % TEI.keywords.ent system 'teikey2.ent'                   >
%TEI.keywords.ent;
<!-- This fragment is used in sec. 3.6.1                          -->
```

Part II

Core Tags and General Rules

Chapter 4

Characters and Character Sets

Computer systems vary greatly in the sets of characters they make available for use in electronic documents; this variety enables users with widely different needs to find computer systems suitable to their work, but it also complicates the interchange of documents among systems; hence the need for a chapter on this topic in these Guidelines.

Four character-set problems arise for the encoder of electronic texts:

1. selecting a character set to use in creating, processing, or storing the electronic text
2. preparing documents for interchange so that the characters within them are not corrupted in transit
3. encoding characters which are not provided by the character set available on the computer system in use
4. signaling shifts from one character set to another, e.g. from the Latin alphabet to Greek and back, or to a special symbol character set and back

This chapter describes the recommended solutions to these problems, in enough detail to satisfy the needs of most users. More detail and more technical information can be found in chapters 28 ('Conformance') on p. 611, 30 ('Rules for Interchange ') on p. 627, 25 ('Writing System Declaration') on p. 571, and 37 ('Obtaining TEI WSDs') on p. 985.

4.1 Local Character Sets

No single character set is prescribed for use in TEI-encoded documents. Users may use any character set available to them, subject to the character set restrictions imposed by the SGML declaration. It is recommended that the character set used be documented by one or more *writing system declarations* (WSD), on which see below. In most cases, a predefined writing system declaration should be suitable. For writing system declarations provided as part of the TEI Guidelines, see chapter 37 ('Obtaining TEI WSDs') on p. 985.

In general, it is most convenient to use a character set readily available on one's computer system, though for special purposes it may be preferable to customize the character set using software specialized for the purpose. Whether to use the usual character set or create a custom set depends on the documents being encoded, the staff support and tools available for customizing the character set, the user's technical facility, etc. A choice must be made between the perceived relative convenience of living with an existing character set and the effort of modifying it to suit one's documents more closely. Care should be taken, in particular where multi-byte character sets are used, that the syntactic distinctions required by an SGML processor are preserved. Where a suitable character set is defined by a national or international standard, local customization should implement the standard rather than inventing yet another non-standard character set. The choice must be made by each individual according to individual circumstances; no general recommendations are made here as to whether locally customized character sets should be used. In principle, however, for local processing, encoders should use whatever character set they find convenient.

4.1.1 Characters Available Locally

When the characters in a text exist in the local character set, the appropriate character codes should be used to represent them. Virtually all computer systems provide at least the 52 letters of the standard Latin alphabet, the ten decimal digits, and some basic punctuation marks.

Other characters, such as Latin characters with diacritics (e.g. ä or é) or non-Latin characters (e.g. Greek, Hebrew, Arabic, Cyrillic) are less universally provided. Oriental scripts pose particular problems because of the size of their character repertoires. If the local character set provides an 'ä' however, there is normally no reason not to use it where that character appears in the text, unless the electronic text is to be moved frequently among machines, in which case one may wish to restrict the electronic text to characters known to translate well among machines. (For more information on moving characters among machines, see section 4.3 ('Character Set Problems in Interchange') on p. 75 below.)

As noted above, full use of a local character set may require that the SGML declaration be modified to define all the characters used as legal SGML characters.

4.1.2 Characters Not Available Locally

Characters not available in the local character set should usually be encoded using SGML *entity references*. In SGML terms, an *entity* (described in more detail in section 2.7 ('SGML Entities ') on p. 29) is any named string of characters, from a single character to an entire system file. Entities are included in SGML documents by *entity references*. Lists of suggested names for all the characters and printers' symbols used by most modern European languages have been published by ISO and others. [1]

For example, the standard entity name for the character 'ä' is 'auml'; a reference to the entity gives the name of the entity, preceded by an ampersand and followed by a semicolon. [2] For example, consider the following German sentence: "Trotz dieser langen Tradition sekundäranalytischer Ansätze wird man die Bilanz tatsächlich durchgeführter sekundäranalytischer Arbeiten aber in wesentlichen Punkten als unbefriedigend empfinden müssen." Using the standard names for a-umlaut and u-umlaut, one could transcribe this sentence thus:

```
Trotz dieser langen Tradition sekund&auml;ranalytischer
Ans&auml;tze wird man die Bilanz tats&auml;chlich
durchgef&uuml;hrter sekund&auml;ranalytischer Arbeiten
aber in wesentlichen Punkten als unbefriedigend
empfinden m&uuml;ssen.
```

As noted above, standard entity names have been defined for most characters used by languages written in the Latin alphabet, and for some other scripts, including Greek, Cyrillic, Coptic, and the International Phonetic Alphabet. A useful subset of these may be found in chapter 37 ('Obtaining TEI WSDs') on p. 985.

Before an entity can be referred to, it must be declared. Standard public entity names can be declared en masse, by including within the DTD subset of the SGML document a reference to the standard public entity which declares them. The German document quoted above, for example, would have the following lines, or their equivalent, in its DTD subset:

```
<!ENTITY % ISOLat1
       PUBLIC "ISO 8879-1986//ENTITIES Added Latin 1//EN">
%ISOLat1;
```

Where no standard entity name exists, or where the standard name is felt unsuitable for some reason, the encoder may declare non-standard entities, using the normal SGML syntax. The replacement string for such entities may vary for different purposes, as in the following extended example.

[1] The most widely used such entity set is to be found in Annex D to ISO 8879; it is also reproduced or summarized in most SGML textbooks, notably Charles F. Goldfarb, *The SGML Handbook* (Oxford: Clarendon Press, 1990). A list of some frequently used standard entity names may be found in chapter 37 ('Obtaining TEI WSDs') on p. 985. Extensive entity sets are being developed by the TEI and others are being documented in the fascicles of ISO/TR 9573: *Technical Report: Information processing — SGML support facilities — Techniques for using SGML* ([Geneva]: ISO, 1988 et seq.).

[2] Strictly speaking the semicolon is not always required; for details see any full treatment of SGML.

In transcribing a manuscript, it might be desirable to distinguish among three distinct forms of the letter 'r'. In the transcript, each of these forms will be encoded by an entity reference, for example: `&, &r2;`, and `&r3;`. Entity declarations must then be provided, within the DTD subset of the SGML document, to define these entities and specify a substitute string.

One possible set of declarations would be as follows:

```
<!ENTITY r  'r[1]'><!-- most common form of 'r'  -->
<!ENTITY r2 'r[2]'><!-- secondary form of 'r'    -->
<!ENTITY r3 'r[3]'><!-- third form of 'r'        -->
```

The expansions shown above will simply flag each occurrence with a number in brackets to indicate which form of 'r' appears in the manuscript.

Another (imaginary) set of declarations might use some printer-specific codes to produce visually distinct versions of the three letters in output:

```
<!ENTITY so '&#27;I&#12;'><!-- shift to alternate printer font -->
<!ENTITY si '&#27;I&#8;'> <!-- shift to basic printer font -->
<!ENTITY r  'r'>             <!-- most common form of 'r'  -->
<!ENTITY r2 '&so;r&si;'>     <!-- secondary form of 'r'    -->
<!ENTITY r3 '&so;&#113;&si;'><!-- third form of 'r'        -->
```

Here the replacement strings for each entity contain *character references*, for example `&#27;`, indicating the decimal value (or *code point*) of the character to be generated by the SGML processor. The assumption is that the sequence of bytes generated will drive a printer to produce a distinctive form of the letter. Such printer instructions are of their nature highly machine- and software-dependent; by allowing them to be confined to a single place (the entity declarations), SGML makes it easier to adapt such device-specific information to new environments.

Alternatively, the declarations can give all three forms the same appearance in the output:

```
<!ENTITY r  'r'><!-- most common form of 'r'  -->
<!ENTITY r2 'r'><!-- secondary form of 'r'    -->
<!ENTITY r3 'r'><!-- third form of 'r'        -->
```

And finally, to ensure that the SGML output uses the same entity references for these characters as does the SGML input, one could use the following declarations.

```
<!ENTITY r  CDATA '&r;' ><!-- most common form of 'r'  -->
<!ENTITY r2 CDATA '&r2;'><!-- secondary form of 'r'    -->
<!ENTITY r3 CDATA '&r3;'><!-- third form of 'r'        -->
```

Wherever locally defined entities are used for the representation of characters in the text, whether to record presentational variants as in this example or to represent special symbols not present in standard character sets or public entity sets, it is recommended that the entities be documented in a writing system declaration; for details see chapter 25 ('Writing System Declaration') on p. 571.

For transcriptions in scripts not supported by the local character set, entity references may prove unwieldy. In such cases, it is also possible to transliterate the material from its original script into the script of the local character set; like a customized local character set, a transliteration scheme should be documented with a writing system declaration. To avoid information loss, a *reversible* transliteration scheme (i.e. one in which it is possible to reconstruct the original writing from the transliteration) should be preferred; wherever possible, standard schemes should be preferred to ad hoc schemes. [3]

For example, using the Beta code transcription developed for ancient Greek by the Thesaurus Linguæ Græcæ,[4] one would transcribe the start of the *Iliad* of Homer thus:

```
*MH=NIN A)/EIDE QEA\ *PHLHI+A/DEW *)AXILH=OS
OU)LOME/NHN, H(\ MURI/' *)AXAIOI=S A)/LGE' E)/QHKE,
```

[3] Reversible transliteration schemes are defined by national and international standards too numerous to list here; another useful source is *ALA-LC Romanization Tables: Transliteration Schemes for Non-Roman Scripts*, approved by the Library of Congress and the American Library Association, tables compiled and edited by Randall K. Barry (Washington: Library of Congress, 1991). Many of these schemes, however, use diacritics which are themselves not always available in standard electronic character sets and thus may require careful adaptation for use in electronic work.

[4] Thesaurus Linguæ Græcæ, *Beta Manual* (Irvine: TLG, [1988]). See also Luci Berkowitz and Karl A. Squitier, *Thesaurus Linguæ Græcæ Canon of Greek Authors and Works* 2nd edition (Oxford: Oxford University Press, 1986).

Some standard transliteration schemes for commonly encoded languages are included in chapter 37 ('Obtaining TEI WSDs') on p. 985.

4.2 Shifting Among Character Sets

Many documents contain material from more than one language: loan words, quotations from foreign languages, etc. Since languages use a variety of *writing systems*, which in turn use a variety of *character repertoires*, shifts in language frequently go hand in hand with shifts in character repertoire and writing system. Since language change is frequently of importance in processing a document, even when no character set shift is needed, the encoding scheme defined here provides a global attribute lang to make it possible to mark language shifts explicitly. This attribute may also be used to trigger character-set shifting by application programs.

Some languages use more than one writing system. For example, some Slavic languages may be written either in the Latin or in the Cyrillic alphabet; some Turkic languages in Cyrillic, Latin, or Arabic script. In such cases, each writing system must be treated separately, as a separate "language". Each distinct value of the lang attribute, therefore, represents a single natural language, a single writing system, and a single coded character set.

Each value used for the lang attribute must correspond to a writing system declaration suitable for the character set and writing system being used. The values should where possible be taken from the standard two- and three-letter language codes defined by the international standard ISO 639.[5] When more than one writing system is used for the same language in the same document, suffixes should be added to the values from ISO 639. A selection of standard language codes, as well as a number of standard writing system declarations, are provided in chapter 37 ('Obtaining TEI WSDs') on p. 985; other writing system declarations may be provided locally.

Like any global attribute, the lang attribute may be used on any element in the SGML document. To mark a technical term, for example, as being in a particular language, one may simply specify the appropriate language on the <term> tag (for which see 13 ('Terminological Databases') on p. 311):

```
<p lang=EN> ...
But then only will there be good ground of hope for the
further advance of knowledge, when there shall be received
and gathered together into natural history a variety of
experiments, which are of no use in themselves, but simply
serve to discover causes and axioms, which I call
<term lang=LA>Experimenta lucifera</term>, experiments of
<term>light</term>, to distinguish them from those which I
call <term lang=LA>fructifera</term>, experiments of
<term>fruit</term>.
<p>Now experiments of this kind have one admirable
property and condition:  they never miss or fail. ...
```

The form in which materials in different writing systems are processed or displayed is not specified by these Guidelines; if appropriate characters are not available locally, application software may choose to display the material in an appropriate transliteration, as a series of entity references, or in other forms. If the local system requires explicit escape sequences or locking-shift control functions to signal shifts to alternate character sets, these escape sequences may be embedded directly in the content of the appropriate element or may be supplied by the application software upon recognition of the language shift. However, escape sequences are vulnerable to loss or misinterpretation when documents are sent from site to site. For this reason, the method of embedding them directly in the document, while allowed within TEI-conformant interchange, is not recommended for general use.[6]

[5] ISO 639: 1988, *Code for the representation of names of languages* ([Geneva]: International Organization for Standardization, 1988). The most recent version of this standard supplies three-letter codes as well as the earlier system of two-letter codes. Either may be used in TEI documents.

[6] Standard methods for character-set shifting are defined in ISO 2022: 1986, *Information processing — ISO 7-bit and 8-bit*

4.3 Character Set Problems in Interchange

Electronic texts may be exchanged over electronic networks, through exchange of magnetic media (e.g. disk or tape), or by other means. In every case except the transmission of storage media from one machine to another machine of the same hardware type running the same operating system and using the same coded character set, the characters are subject to translation and interpretation, and hence to misinterpretation and corruption, by utility software working somewhere on the interchange path. Network gateways, tape-reading software, and disk utilities routinely translate from one character set to another before passing the data on. If the utility errs in identifying the character set, or if several utilities translate back and forth among character sets using non-reversible translations, the chances are good that characters will be garbled and information lost.

At this time (1994), it is recommended that in texts subject to "blind" interchange (that is, interchange between parties who do not or cannot make explicit agreements over the character set to be used in interchange) no characters should be used other than those in the international reference version (IRV) of ISO 646.[7] Other characters should be represented by SGML entity references. For scripts (such as those of East Asia) which require a character repertoire larger than 255 characters, regional proposals for character and entity sets to be used in non-negotiated interchange must be developed and published internationally.

In some cases, even the characters in ISO 646 IRV are corrupted in transit; where interchange takes place over links not reliable for the full IRV character set, the characters subject to misinterpretation and corruption should be replaced by standard entities. At present, the characters least susceptible to loss or misinterpretation in transit among systems are those shown below, which represent a subset of the characters in IRV and may thus be called the *ISO 646 subset*.

```
a b c d e f g h i j k l m n o p q r s t u v w x y z
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
0 1 2 3 4 5 6 7 8 9
" % & ' ( ) * + , - . / : ; < = > ? _
```

In interchange over any transmission link, the transmitted document should contain only those characters which safely survive transmission over the link; others should be represented with entity references, or with transliterations, as described above.

In blind interchange by means of magnetic or optical media, it is recommended that the document be encoded using some well documented and widely used standard character set.

In blind interchange over networks, it is recommended that the transmitted document contain only characters known to travel safely over the networks involved. In the most general case, those characters are the ISO 646 subset given above.

The problem of character data loss in document interchange is in principle a temporary one; eventually, it should disappear as improvements are made in network handling of character data and as more networks support larger character sets such as ISO 10646.[8] The need for backward compatibility with older systems is likely, however, to ensure that care must be taken in blind interchange for the foreseeable future.

---

coded character sets — Code extension techniques, 3d ed. ([Geneva]: International Organization for Standardization, 1986). A standard set of control functions, including methods of specifying script direction (left-right, right-left, top-down, etc.), is defined by ISO 6429: 1992, *Information processing — Control functions for 7-bit and 8-bit coded character sets*, ([Geneva]: ISO, 1992). These and related standards are recommended to the notice of those designing systems to handle multiple character sets.

[7] ISO (International Organization for Standardization). *ISO 646: 1991. Information processing — ISO 7-bit coded character set for information interchange.* ([Geneva]: International Organization for Standardization, 1991.)

[8] ISO (International Organization for Standardization), and IEC (International Electrotechnical Commission), *ISO/IEC 10646-1: 1993. Information technology — Universal Multiple-Octet Coded Character Set (UCS) — Part 1: Architecture and Basic Multilingual Plane.* ([Geneva]: International Organization for Standardization, 1993.) The Basic Multilingual Plane of ISO 10646 is identical to the sixteen-bit character set known as Unicode: The Unicode Consortium, *The Unicode Standard: Worldwide Character Encoding* Version 1.0, Volume 1 (Reading, Mass.: Addison-Wesley, 1991). Version 1.0 of Unicode has been superseded by later work incorporated into ISO/IEC 10646; these changes are be reflected in the official published standard, and will be documented in version 1.1 of Unicode.

For further discussion of negotiated and non-negotiated interchange, see chapter 30 (‘Rules for Interchange ’) on p. 627.

4.4 The Writing System Declaration

Each language and writing system used in a document should be documented in a Writing System Declaration (WSD), which specifies:

- a formal name for the writing system and language, for use as a `lang` attribute value on elements encoded in it
- a specification for the meaning of each character available in the writing system

The characters available in a writing system may be specified in the WSD for that writing system in one or more of the following ways:

- by reference to an international, national, or TEI-registered coded character set or entity set
- by reference to such a standard followed by formal declaration of all exceptions
- by providing a formal declaration for each character used

Individual characters within a WSD are formally declared, where necessary, by providing the following information:

- the unique code used to represent the character
- special properties such as whether the character is a diacritic mark or not
- brief textual description of the character
- standard or local entity name used for the character in interchange
- other standard identifiers for the character, if available, such as its code in the “Universal Character Set” of ISO 10646 or Unicode and its standard number in the Association for Font Information Interchange registry.
- optionally, some specification of a suitable graphic rendition for the character in a suitable notation (e.g. graphic image, Metafont program, etc.)

The writing system declaration is one of a set of *auxiliary* documents which provide documentation relevant to the processing of TEI texts. Auxiliary documents are themselves SGML documents, for which document type declarations are provided. The DTD for the Writing System Declaration is discussed in detail in chapter 25 (‘Writing System Declaration’) on p. 571. Standard Writing System Declarations are provided in chapter 37 (‘Obtaining TEI WSDs’) on p. 985.

Chapter 5

The TEI Header

This chapter addresses the problems of describing an encoded work so that the text itself, its source, its encoding, and its revisions are all thoroughly documented. Such documentation is equally necessary for scholars using the texts, for software processing them, and for cataloguers in libraries and archives. Together these descriptions and declarations provide an electronic analogue to the title page attached to a printed work. They also constitute an equivalent for the content of the code books or introductory manuals customarily accompanying electronic data sets.

Every TEI-conformant text must carry such a set of descriptions, prefixed to it and encoded as described in this chapter. The set is known as the *TEI header*, tagged <teiHeader>, and it has four major parts:

- a *file description*, tagged <fileDesc>, containing a full bibliographical description of the computer file itself, from which a user of the text could derive a proper bibliographic citation, or which a librarian or archivist could use in creating a catalogue entry recording its presence within a library or archive. The file description also includes information about the source or sources from which the electronic text was derived. The TEI elements used to encode a file description are described in section 5.2 ('The File Description') on p. 80 below.
- an *encoding description*, tagged <encodingDesc>, which describes the relationship between an electronic text and its source or sources. It allows for detailed description of whether (or how) the text was normalized during transcription, how the encoder resolved ambiguities in the source, what levels of encoding or analysis were applied, and similar matters. The TEI elements used to encode the encoding description are described in section 5.3 ('The Encoding Description') on p. 93 below.
- a *text profile*, tagged <profileDesc>, containing classificatory and contextual information about the text, such as its subject matter, the situation in which it was produced, the individuals described by or participating in producing it, and so forth. Such a text profile is of particular use in highly structured composite texts such as corpora or language collections, where it is often highly desirable to enforce a controlled descriptive vocabulary or to perform retrievals from a body of text in terms of text type or origin. The text profile may however be of use in any form of automatic text processing. The TEI elements used to encode the profile description are described in section 5.4 ('The Profile Description') on p. 108 below.
- a *revision history*, tagged <revisionDesc>, which allows the encoder to provide a history of changes made during the development of the electronic text. The revision history is important for *version control* and for resolving questions about the history of a file. The TEI elements used to encode the revision description are described in section 5.5 ('The Revision Description') on p. 112 below.

A TEI header can be a very large and complex object, or it may be a very simple one. Some application areas (for example, the construction of language corpora and the transcription of spoken texts) will require more specialized and detailed information than others. The present proposals therefore define both a *core* set of elements, (all of which may be used without formality in any TEI header) and *additional tagsets*, which may be invoked as extensions as needed. For more

details of this extension mechanism, see chapter 3.2 ('Core, Base, and Additional Tag Sets') on p. 36; the header extensions are fully described in chapter 23 ('Language Corpora') on p. 537, which should be read in conjunction with the present chapter.

The next section of the present chapter briefly introduces the overall structure of the header, and the kinds of data it may contain. This is followed by a detailed description of all the constituent elements which may be used in the core header. Section 5.6 ('Minimal and Recommended Headers ') on p. 114, at the end of the present chapter, discusses the recommended content of a minimal TEI header, and its relation to standard library cataloguing practices. Recommendations relevant to the use of TEI headers as free-standing documents, for interchange among libraries, data archives, and similar institutions may be found in chapter 24 ('The Independent Header') on p. 559.

5.1 Organization of the TEI Header

5.1.1 The TEI Header and Its Components

The <teiHeader> element should be clearly distinguished both from the *SGML prolog*, which comprises the SGML declaration and the document type declaration (see chapter 2 ('A Gentle Introduction to SGML ') on p. 15), and from the *front matter* of the text itself (for which see section 7.4 ('Front Matter') on p. 201). A composite text, such as a corpus or collection, may contain several headers, as further discussed below. In the usual case however, a TEI-conformant text will contain a single <teiHeader> element, followed by a single <text> element.

The header element has the following description:

<teiHeader> supplies the descriptive and declarative information making up an "electronic title page" prefixed to every
TEI-conformant text. Attributes include:

type specifies the kind of document to which the header is attached. Legal values are:

text the header is attached to a single text.

corpus the header is attached to a corpus.

As discussed above, the <teiHeader> element has four principal components:

<fileDesc> contains a full bibliographic description of an electronic file.

<encodingDesc> documents the relationship between an electronic text and the source or sources from which it was derived.

<profileDesc> provides a detailed description of non-bibliographic aspects of a text, specifically the languages and sublanguages used, the situation in which it was produced, the participants and their setting.

<revisionDesc> summarizes the revision history for a file.

Of these, only the <fileDesc> element is required in all TEI headers; the others are optional. The full form of a TEI header is thus:

```
<teiHeader>
    <fileDesc> ... </fileDesc>
    <encodingDesc> ... </encodingDesc>
    <profileDesc> ... </profileDesc>
    <revisionDesc> ... </revisionDesc>
</teiHeader>
```

while a minimal header takes the form:

```
<teiHeader>
    <fileDesc> ... </fileDesc>
</teiHeader>
```

In the case of language corpora or collections, it may be desirable to record header information either at the level of individual components in the corpus or collection, or once for all at the

level of the corpus or collection itself, or at both levels. More details concerning the tagging of composite texts are given in section 23 ('Language Corpora') on p. 537, which should be read in conjunction with the current chapter. An optional type attribute may also be supplied on the <teiHeader> element to indicate whether the header applies to a corpus or a single text. A corpus may thus take the form:

```
<tei.2>
    <teiHeader type=corpus> ...
    <!-- header for corpus-level information -->
    </teiHeader>
    <tei.2>
         <teiHeader type=text> ...
         <!-- header for text-level information -->
         </teiHeader>
         <text> ... </text>
    </tei.2>
    <tei.2>
         <teiHeader type=text> ... </teiHeader>
         <text> ... </text>
    </tei.2>
    <!-- etc. -->
</tei.2>
```

The tags required for the TEI header are defined in the DTD file *teihdr2* which first defines the <teiHeader> element:

```
<!-- 5.1.1:  The TEI Header                                  -->
<!-- teihdr2.dtd  Tags for TEI Header.                       -->

<!-- Text Encoding Initiative: Guidelines for Electronic     -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.   -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy   -->
<!-- in any form is granted, provided this notice is         -->
<!-- included in all copies.                                 -->

<!-- These materials may not be altered; modifications to    -->
<!-- these DTDs should be performed as specified in the      -->
<!-- Guidelines in chapter "Modifying the TEI DTD."          -->

<!-- These materials subject to revision. Current versions   -->
<!-- are available from the Text Encoding Initiative.        -->
<!ELEMENT teiHeader     - -  (fileDesc, encodingDesc*,
                              profileDesc*, revisionDesc?)      >
<!ATTLIST teiHeader          %a.global;
          type               CDATA               text
          creator            CDATA               #IMPLIED
          status             (new | update)      new
          date.created       CDATA               #IMPLIED
          date.updated       CDATA               #IMPLIED      >
<!-- (continued in sec. 5.1.1)                               -->
```

Then it defines the rest of the header elements, embedding the DTD fragments found later in this chapter:

```
<!-- 5.1.1:  The TEI Header (cont'd)                         -->
<!-- (continuation of sec. 5.1.1)                            -->
<!-- ... declarations from section 5.2                       -->
<!--     (The file description)                              -->
<!--     go here ...                                         -->
<!-- ... declarations from section 5.2.7                     -->
<!--     (The source description)                            -->
<!--     go here ...                                         -->
```

```
<!-- ... declarations from section 5.3                    -->
<!--     (The encoding description)                       -->
<!--     go here ...                                      -->
<!-- ... declarations from section 5.4                    -->
<!--     (The profile description)                        -->
<!--     go here ...                                      -->
<!-- ... declarations from section 5.5                    -->
<!--     (The Revision Description)                       -->
<!--     go here ...                                      -->
```

5.1.2 Types of Content in the TEI Header

The elements occurring within the TEI header may contain several types of content; the following list indicates how these types of content are described in the following sections:

free prose Most elements contain simple running prose at some level. Many elements may contain either prose (possibly organized into paragraphs) or more specific elements, which themselves contain prose. In this chapter's descriptions of element content, the phrase 'prose description' should be understood to imply a series of paragraphs, each marked with the <p> tag. The word 'phrase', by contrast, should be understood to imply character data, interspersed as need be with phrase-level elements, but not organized into paragraphs. For more information on paragraphs, highlighted phrases, lists, etc., see section 6.1 ('Paragraphs') on p. 120.

grouping elements Elements whose names end with the suffix 'Stmt' (e.g. <editionStmt>, <situationStmt>) usually enclose a group of specialized elements recording some structured information. In the case of the bibliographic elements, the suffix 'Stmt' is used in names of elements corresponding to the "areas" of the International Standard Bibliographic Description. [1]

In most cases grouping elements may contain prose descriptions as an alternative to the set of specialized elements, thus allowing the encoder to choose whether or not the information concerned should be presented in a structured form or in prose.

declarations Elements whose names end with the suffix 'Decl' (e.g. <subjectDecl>, <refsDecl>) enclose information about specific encoding practices applied in the electronic text; often these practices are described in coded form. Typically, such information takes the form of a series of declarations, identifying a code with some more complex structure or description. A declaration which applies to more than one text or division of a text need not be repeated in the header of each such text. Instead, the decls attribute of each text (or subdivision of the text) to which the sampling declaration applies may be used to supply a cross reference to it, as further described in section 23.3 ('Associating Contextual Information with a Text') on p. 550.

descriptions Elements whose name end with the suffix 'Desc' (e.g. <settingDesc>, <projectDesc>) contain a prose description, possibly organized under some specific headings by suggested sub-elements, but not necessarily so.

5.2 The File Description

This section describes the <fileDesc> element, which is the first component of the <teiHeader> element.

The bibliographic description of a machine-readable text resembles in structure that of a book, an article, or any other kind of textual object. The file description element of the TEI header has therefore been closely modelled on existing standards in library cataloguing; it should thus provide enough information to allow users to give standard bibliographic references to the electronic text, and to allow cataloguers to catalogue it. Bibliographic citations occurring elsewhere in the header, and also in the text itself, are derived from the same model (on bibliographic

[1] On the relation between the TEI proposals and other standards for bibliographic description, see further section 5.7 ('Note for Library Cataloguers') on p. 116.

citations in general, see further section 6.10 ('Bibliographic Citations and References') on p. 162). See further section 5.7 ('Note for Library Cataloguers') on p. 116.

The bibliographic description of the electronic text (not its source) is given in the mandatory <fileDesc> element:

<fileDesc> contains a full bibliographic description of an electronic file.

The <fileDesc> element contains three mandatory elements and four optional elements, each of which is described in more detail in sections 5.2.1 ('The Title Statement') on p. 82 to 5.2.6 ('The Notes Statement') on p. 88 below. These elements are listed below in the order in which they must be given within the <fileDesc> element.

<titleStmt> groups information about the title of a work and those responsible for its intellectual content.

<editionStmt> groups information relating to one edition of a text.

<extent> describes the approximate size of the electronic text as stored on some carrier medium, specified in any convenient units.

<publicationStmt> groups information concerning the publication or distribution of an electronic or other text.

<seriesStmt> groups information about the *series*, if any, to which a publication belongs.

<notesStmt> collects together any notes providing information about a text additional to that recorded in other parts of the bibliographic description.

<sourceDesc> supplies a bibliographic description of the copy text(s) from which an electronic text was derived or generated.

A file description containing all possible subelements has the following structure:

```
<teiHeader>
    <fileDesc>
         <titleStmt> ... </titleStmt>
         <editionStmt> ... </editionStmt>
         <extent> ... </extent>
         <publicationStmt> ... </publicationStmt>
         <seriesStmt> ... </seriesStmt>
         <notesStmt> ... </notesStmt>
         <sourceDesc> ... </sourceDesc>
    </fileDesc>
    <!-- remainder of TEI Header here -->
</teiHeader>
```

Several of these elements may be omitted; a minimal file description has the following structure:

```
<teiHeader>
    <fileDesc>
         <titleStmt> ... </titleStmt>
         <publicationStmt> ... </publicationStmt>
         <sourceDesc> ... </sourceDesc>
    </fileDesc>
    <!-- remainder of TEI Header here -->
</teiHeader>
```

The <fileDesc> itself has the following formal definition:

```
<!-- 5.2:  The file description                                  -->
<!ELEMENT fileDesc      - -  (titleStmt, editionStmt?, extent?,
                              publicationStmt, seriesStmt?,
                              notesStmt?, sourceDesc+ )          >
<!ATTLIST fileDesc           %a.global;                          >
<!-- ... declarations from section 5.2.1                        -->
<!--     (The title statement)                                  -->
<!--     go here ...                                            -->
<!-- ... declarations from section 5.2.2                        -->
<!--     (The edition statement)                                -->
```

```
<!--     go here ...                                    -->
<!-- ... declarations from section 5.2.3                -->
<!--     (The extent statement)                         -->
<!--     go here ...                                    -->
<!-- ... declarations from section 5.2.4                -->
<!--     (The publication statement)                    -->
<!--     go here ...                                    -->
<!-- ... declarations from section 5.2.5                -->
<!--     (The series statement)                         -->
<!--     go here ...                                    -->
<!-- ... declarations from section 5.2.6                -->
<!--     (The notes statement)                          -->
<!--     go here ...                                    -->
<!-- This fragment is used in sec. 5.1.1                -->
```

5.2.1 The Title Statement

The <titleStmt> element is the first component of the <fileDesc> element, and is mandatory:

<titleStmt> groups information about the title of a work and those responsible for its intellectual content.

It contains the title given to the electronic work, together with optionally one or more *statements of responsibility* which identify the encoder, author, compiler, or other parties responsible for it:

<title> contains the title of a work, whether article, book, journal, or series, including any alternative titles or subtitles.

<author> in a bibliographic reference, contains the name of the author(s), personal or corporate, of a work; the primary *statement of responsibility* for any bibliographic item.

<sponsor> specifies the name of a sponsoring organization or institution.

<funder> specifies the name of an individual, institution, or organization responsible for the funding of a project or text.

<principal> supplies the name of the principal researcher responsible for the creation of an electronic text.

<respStmt> supplies a statement of responsibility for someone responsible for the intellectual content of a text, edition, recording, or series, where the specialized elements for authors, editors, etc. do not suffice or do not apply.

<resp> contains a phrase describing the nature of a person's intellectual responsibility.

<name> contains a proper noun or noun phrase.

The <title> element contains the chief name of the file, including any alternative title or subtitles it may have. It may be repeated, if the file has more than one title, (perhaps in different languages) and takes whatever form is considered appropriate by its creator. Where the electronic work is derived from an existing source text, it is strongly recommended that the title for the former should also be derived from the latter, but that it should be clearly distinguishable from it. For example, do not call the computer file "A Sanskrit-English Dictionary, based upon the St. Petersburg Lexicons". Call it, rather, "Sanskrit-English Dictionary, based upon the St. Petersburg Lexicons: a machine readable transcription". If you wish to retain some or all of the title of the source text in the title of the computer file, then introduce one of the following phrases:

title of source : a machine readable transcription.
title of source : electronic edition.

- A machine readable version of: [title of source].

This will distinguish the computer file from the source text in citations and in catalogues which contain descriptions of both types of material.

The computer file will almost certainly have an external name (its "filename" or "data set name") or reference number on the computer system where it resides at any time. This name is likely to change frequently, as new copies of the file are made on the computer system. Its form

is entirely dependent on the particular computer system in use and thus cannot always easily be transferred from one system to another. For these reasons, these Guidelines strongly recommend that such names should *not* be used as the title for any computer file.

Helpful guidance on the formulation of useful descriptive titles in difficult cases may be found in the Anglo-American Cataloguing Rules (AACR 2), chapter 25, or in equivalent national-level bibliographical documentation.[2]

The specialized elements <author>, <sponsor>, <funder>, and <principal>, and the more general <respStmt> provide the *statements of responsibility* which identify the persons responsible for the intellectual or artistic content of an item and any corporate bodies from which it emanates.

Any number of statements of responsibility may occur within the title statement. At a minimum, identify the author of the text and the creator of the machine-readable file. If the bibliographic description is for a corpus, identify the creator of the corpus. These identifications are mandatory when applicable, though not enforceable by the SGML parser. Optionally include also names of others involved in the transcription or elaboration of the text, sponsors, and funding agencies. The name of the person responsible for physical data input need not normally be recorded, unless that person is also intellectually responsible for some aspect of the creation of the file.

Where the person whose responsibility is to be documented is not an author, sponsor, funding body or principal researcher, the <respStmt> element should be used. This has two subcomponents: a <name> element identifying a responsible individual or organization, and a <resp> element indicating the nature of the responsibility. No specific recommendations are made at this time as to appropriate content for the <resp>: it should make clear the nature of the responsibility concerned, as in the examples below.

Names given may be personal names or corporate names. Give all names in the form in which the persons or bodies wish to be publicly cited. This would usually be the fullest form of the name, including first names. [3] Examples:

```
<titleStmt>
  <title>Capgrave's Life of St. John Norbert:  a
     machine-readable transcription</title>
  <respStmt>
      <resp>compiled by</resp> <name>P.J. Lucas</name>
  </respStmt>
</titleStmt>
```

```
<titleStmt>
  <title>Two stories by Edgar Allen Poe:
           electronic version</title>
  <author>Poe, Edgar Allen (1809-1849)
  <respStmt><resp>compiled by</resp>
  <name>James D. Benson</name></respStmt>
</titleStmt>
```

```
<titleStmt>
  <title>Yogadar&sacute;anam (arth&amacr;t
         yogas&umacr;trap&macr;&tdot;ha&hdot;):
         a machine readable transcription.</title>
  <title>The Yogas&umacr;utras of Pata&ntilde;jali:
         a machine readable transcription.</title>
  <funder>Wellcome Institute for the History of Medicine
  </funder>
  <principal>Dominik Wujastyk</principal>
  <respStmt><name>Wieslaw Mical</name>
        <resp>data entry and proof correction</resp>
  </respStmt>
```

[2] Michael Gorman and Paul W. Winkler, eds., *Anglo-American Cataloguing Rules*, Second Edition (Chicago: American Library Association; London: Library Association; Ottawa: Canadian Library Association, 1978).

[3] Agencies compiling catalogues of machine-readable files are recommended to use available authority lists, such as the Library of Congress Name Authority List, for all common personal names.

```
  <respStmt><name>Jan Hajic</name>
        <resp>conversion to TEI-conformant markup</resp>
  </respStmt>
</titleStmt>
```

The formal definition of the <titleStmt> element and its constituents is as follows:

```
<!-- 5.2.1:  The title statement                                    -->
<!ELEMENT titleStmt     - o  ((title+, (author | editor |
                              sponsor | funder | principal |
                              respStmt)*))                          >
<!ATTLIST titleStmt          %a.global;                             >
<!ELEMENT sponsor       - o  ( %phrase.seq; )                       >
<!ATTLIST sponsor            %a.global;                             >
<!ELEMENT funder        - o  ( %phrase.seq; )                       >
<!ATTLIST funder             %a.global;                             >
<!ELEMENT principal     - o  (%phrase.seq;)                         >
<!ATTLIST principal          %a.global;                             >
<!-- The TITLE, AUTHOR, NAME, RESPSTMT, and RESP elements      -->
<!-- are declared in file teicore2.dtd, not here.               -->

<!-- This fragment is used in sec. 5.2                          -->
```

5.2.2 The Edition Statement

The <editionStmt> element is the second component of the <fileDesc> element. It is optional but recommended.

<editionStmt> groups information relating to one edition of a text.

It contains either phrases or more specialized elements identifying the edition and those responsible for it:

<edition> describes the particularities of one edition of a text.

<respStmt> supplies a statement of responsibility for someone responsible for the intellectual content of a text, edition, recording, or series, where the specialized elements for authors, editors, etc. do not suffice or do not apply.

<name> contains a proper noun or noun phrase.

<resp> contains a phrase describing the nature of a person's intellectual responsibility.

For printed texts, the word 'edition' applies to the set of all the identical copies of an item produced from one master copy and issued by a particular publishing agency or a group of such agencies. A change in the identity of the distributing body or bodies does not normally constitute a change of edition, while a change in the master copy does.

For electronic texts, the notion of a "master copy" is not entirely appropriate, since they are far more easily copied and modified than printed ones; nonetheless the term 'edition' may be used for a particular state of a machine-readable text at which substantive changes are made and fixed. Synonymous terms used in these Guidelines are 'version,' 'level,' and 'release'. The words 'revision' and 'update', by contrast, are used for minor changes to a file which do not amount to a new edition.

No simple rule can specify how "substantive" changes have to be before they are regarded as producing a new edition, rather than a simple update. The general principle proposed here is that the production of a new edition entails a significant change in the intellectual content of the file, rather than its encoding or appearance. The addition of analytic coding to a text would thus constitute a new edition, while automatic conversion from one coded representation to another would not. Changes relating to the character code or physical storage details, corrections of misspellings, simple changes in the arrangement of the contents and changes in the output format do not normally constitute a new edition. The addition of new information (e.g. a linguistic analysis expressed in part-of-speech tagging, sound or graphics, referential links to external datasets) almost always does constitute a new edition.

Clearly, there are always border line cases and the matter is somewhat arbitrary. The simplest rule is: if you think that your file is a new edition, then call it such. An edition statement is optional for the first release of a machine-readable file; it is mandatory for each later release, though this requirement cannot be enforced by the SGML parser.

Note that *all* changes in a file, whether or not they are regarded as constituting a new edition or simply a new revision, should be independently noted in the revision description section of the file header (see section 5.5 ('The Revision Description') on p. 112).

The <edition> element should contain phrases describing the edition or version, including the word 'edition', 'version', or equivalent, together with a number or date, or terms indicating difference from other editions such as 'new edition', 'revised edition' etc. Any dates that occur within the edition statement should be marked with the <date> element. The n attribute of the <edition> element may be used as elsewhere to supply any formal identification (such as a version number) for the edition.

One or more <respStmt> elements may also be used to supply statements of responsibility for the edition in question. These may refer to individuals or corporate bodies and can indicate functions such as that of a reviser, or can name the person or body responsible for the provision of supplementary matter, of appendices, etc., in a new edition. For further detail on the <respStmt> element, see section 6.10 ('Bibliographic Citations and References') on p. 162.

Some examples follow:

```
<editionStmt>
  <edition n='P2'>Second draft, substantially
       extended, revised, and corrected.</edition>
</editionStmt>
```

```
<editionStmt>
<edition>Student's edition, <date>June 1987</date></edition>
<respStmt>
  <resp>New annotations by</resp> <name>George Brown</name>
</respStmt>
</editionStmt>
```

The formal definition of the <editionStmt> element is as follows:

```
<!-- 5.2.2:  The edition statement                              -->
<!ELEMENT editionStmt   - o  ( (edition, respStmt*) | p+ )      >
<!ATTLIST editionStmt        %a.global;                         >
<!ELEMENT edition       - o  (%phrase.seq;)                     >
<!ATTLIST edition            %a.global;                         >
<!-- This fragment is used in sec. 5.2                         -->
```

5.2.3 Type and Extent of File

The <extent> element is the third component of the <fileDesc> element. It is optional.

<extent> describes the approximate size of the electronic text as stored on some carrier medium, specified in any convenient units.

For printed books, information about the carrier, such as the kind of medium used and its size, are of great importance in cataloguing procedures. The print-oriented rules for bibliographic description of an item's medium and extent need some re-interpretation when applied to electronic media. An electronic file exists as a distinct entity quite independently of its carrier and remains the same intellectual object whether it is stored on a magnetic tape, a CD-ROM, a set of floppy disks or as a file on a mainframe computer. Since, moreover, these Guidelines are specifically aimed at facilitating transparent document storage and interchange, any purely machine-dependent information should be irrelevant as far as the file header is concerned.

This is particularly true of information about *file-type* although library-oriented rules for cataloguing often distinguish two types of computer file: "data" and "programs". This distinction is quite difficult to draw in some cases, for example, hypermedia or texts with built in search and retrieval software. However, since all files covered by these Guidelines are of the same kind, SGML representations, it is unnecessary to specify the file type separately.

Although it is equally system-dependent, some measure of the size of the computer file may be of use for cataloguing and other practical purposes. Because the measurement and expression of file size is fraught with difficulties, only very general recommendations are possible; the element <extent> is provided for this purpose. It contains a phrase indicating the size or approximate size of the computer file in one of the following ways:

- in bytes of a specified length (e.g. "4000 16-bit bytes")
- as falling within a range of categories, for example:
  - less than 1 Mb
  - between 1 Mb and 5 Mb
  - between 6 Mb and 10 Mb
  - over 10 Mb
- in terms of any convenient logical units (for example, words or sentences, citations, paragraphs)
- in terms of any convenient physical units (for example, blocks, disks, tapes)

Examples:

```
<extent>between 1 16-bit Mb and 2 16-bit Mb</extent>
<extent>4532 bytes</extent>
<extent>3200 sentences</extent>
<extent>5 3.5" High Density Diskettes</extent>
```

The <extent> element has the following formal declaration:

```
<!-- 5.2.3:  The extent statement                              -->
<!ELEMENT extent        - o  (%phrase.seq; )                    >
<!ATTLIST extent             %a.global;                         >
<!-- This fragment is used in sec. 5.2                         -->
```

5.2.4 Publication, Distribution, etc.

The <publicationStmt> element is the fourth component of the <fileDesc> element and is mandatory.

<publicationStmt> groups information concerning the publication or distribution of an electronic or other text.

It may contain either a simple prose description, or groups of the elements described below:

<publisher> provides the name of the organization responsible for the publication or distribution of a bibliographic item.

<distributor> supplies the name of a person or other agency responsible for the distribution of a text.

<authority> supplies the name of a person or other agency responsible for making an electronic file available, other than a publisher or distributor.

The *publisher* is the person or institution by whose authority a given edition of the file is made public. The *distributor* is the person or institution from whom copies of the text may be obtained. Where a text is not considered formally published, but is nevertheless made available for circulation by some individual or organization, this person or institution is termed the *release authority*.

At least one of these three elements must be present, unless the entire publication statement is given as prose. Each may be followed by one or more of the following elements, in the following order:

<pubPlace> contains the name of the place where a bibliographic item was published.

<address> contains a postal or other address, for example of a publisher, an organization, or an individual.

<idno> supplies any standard or non-standard number used to identify a bibliographic item. Attributes include:

type categorizes the number, for example as an ISBN or other standard series.

<availability> supplies information about the availability of a text, for example any restrictions on its use or distribution, its copyright status, etc. Attributes include:

status supplies a code identifying the current availability of the text. Sample values include:

free the text is freely available.

unknown the status of the text is unknown.

restricted the text is not freely available.

<date> contains a date in any format.

Note that the dates, places, etc., given in the publication statement relate to the publisher, distributor, or release authority most recently mentioned. If the text was created at some date other than its date of publication, its date of creation should be given within the <profileDesc> element, not in the publication statement. Give any other useful dates (e.g., dates of collection of data) in a note.

Additional detailed tagsets may be used for the encoding of names, dates and addresses, as further described in section 6.4 ('Names, Numbers, Dates, Abbreviations, and Addresses') on p. 132 and chapter 20 ('Names and Dates') on p. 487.

Examples:

```
<publicationStmt>
     <publisher>Oxford University Press</publisher>
     <pubPlace>Oxford</pubPlace> <date>1989</date>
     <idno type=ISBN>0-19-254705-4</idno>
     <availability><p>Copyright 1989, Oxford University Press
     </availability>
</publicationStmt>

<publicationStmt>
    <authority>James D. Benson</authority>
    <pubPlace>London</pubPlace> <date>1984</date>
</publicationStmt>

<publicationStmt>
     <publisher>Sigma Press</publisher>
     <date>1991</date>
     <address>21 High Street, Wilmslow, Cheshire M24 3DF</address>
     <distributor>Oxford Text Archive</distributor>
     <idno type='ota'>1256
     <availability><p>Available with prior consent of depositor for
        purposes of academic research and teaching only.
</publicationStmt>
```

The publication statement and its components are formally defined as follows:

```
<!-- 5.2.4:  The publication statement                          -->
<!ELEMENT publicationStmt
                        - o  (p+ | ( (publisher | distributor |
                             authority) & (pubPlace?, address?,
                             idno*, availability?, date?)+ )+ )
                                                                >
<!ATTLIST publicationStmt    %a.global;                         >
<!ELEMENT distributor    - o  (%phrase.seq;)                    >
<!ATTLIST distributor        %a.global;                         >
<!ELEMENT authority      - o  (%phrase.seq;)                    >
<!ATTLIST authority          %a.global;                         >
<!ELEMENT idno           - o  (#PCDATA)                         >
<!ATTLIST idno               %a.global;
          type               CDATA               #IMPLIED      >
<!ELEMENT availability   - o  (p+)                              >
<!ATTLIST availability       %a.global;
          status             (free | unknown | restricted)
                                                 #IMPLIED      >
<!-- The PUBLISHER, PUBPLACE, and ADDRESS elements are        -->
<!-- defined in file teicore2.dtd.                            -->

<!-- This fragment is used in sec. 5.2                        -->
```

5.2.5 The Series Statement

The <seriesStmt> element is the fifth component of the <fileDesc> element and is optional.

<seriesStmt> groups information about the *series*, if any, to which a publication belongs.

In bibliographic parlance, a *series* may be defined in one of the following ways:

- A group of separate items related to one another by the fact that each item bears, in addition to its own title proper, a collective title applying to the group as a whole. The individual items may or may not be numbered.
- Each of two or more volumes of essays, lectures, articles, or other items, similar in character and issued in sequence.
- A separately numbered sequence of volumes within a series or serial.

The <seriesStmt> element may contain a prose description or one or more of the following more specific elements:

<title> contains the title of a work, whether article, book, journal, or series, including any alternative titles or subtitles.
<idno> supplies any standard or non-standard number used to identify a bibliographic item.
<respStmt> supplies a statement of responsibility for someone responsible for the intellectual content of a text, edition, recording, or series, where the specialized elements for authors, editors, etc. do not suffice or do not apply.
<resp> contains a phrase describing the nature of a person's intellectual responsibility.
<name> contains a proper noun or noun phrase.

The <idno> may be used to supply any identifying number associated with the item, including both standard numbers such as an ISSN and particular issue numbers.[4] Its type attribute is used to categorize the number further, taking the value ISSN for an ISSN for example.

Examples:

```
<seriesStmt>
    <seriesTitle>Machine-Readable Texts for the Study of
    Indian Literature</seriesTitle>
    <respStmt>
         <resp>ed. by</resp> <name>Jan Gonda</name>
     </respStmt>
     <idno type="vol">1.2</idno>
     <idno type ='ISSN'>0 345 6789</idno>
</seriesStmt>
```

The series.statement has the following formal definition:

```
<!-- 5.2.5:  The series statement                              -->
<!ELEMENT seriesStmt    - O  ( (title, (idno | respStmt)*) | p+
                             )                                  >
<!ATTLIST seriesStmt         %a.global;                         >
<!-- This fragment is used in sec. 5.2                         -->
```

Its components are all defined elsewhere.

5.2.6 The Notes Statement

The <notesStmt> element is the sixth component of the <fileDesc> element and is optional. If used, it contains one or more <note> elements, each containing a single piece of descriptive information of the kind treated as "general notes" in traditional bibliographic descriptions.

<notesStmt> collects together any notes providing information about a text additional to that recorded in other parts of the bibliographic description.
<note> contains a note or annotation.

[4] Arabic numerals separated by punctuation are recommended for this purpose (e.g., 6.19.33, not VI/xix:33).

Some information found in the notes area in conventional bibliography has been assigned specific elements in these Guidelines; in particular the following items should be tagged as indicated, rather than as general notes:

- the nature, scope, artistic form or purpose of the file; also the genre or other intellectual category to which it may belong: e.g. "Text types: newspaper editorials and reportage, science fiction, westerns, and detective stories". These should be formally described within the <profileDesc> element (section 5.4 ('The Profile Description') on p. 108).
- summary description providing a factual, non-evaluative account of the subject content of the file. E.g. "Transcribes interviews on general topics with native speakers of English in 17 cities during the spring and summer of 1963." These should also be formally described within the <profileDesc> element (section 5.4 ('The Profile Description') on p. 108).
- bibliographic details relating to the source or sources of an electronic text: e.g. "Transcribed from the Norton facsimile of the 1623 Folio". These should be formally described in the <sourceDesc> element (section 5.2.7 ('The Source Description') on p. 90).
- further information relating to publication, distribution or release of the text, including sources from which the text may be obtained, any restrictions on its use or formal terms on its availability. These should be placed in the appropriate division of the <publicationStmt> element (section 5.2.4 ('Publication, Distribution, etc') on p. 86).
- publicly documented numbers associated with the file: e.g. "ICPSR study number 1803" or "Oxford Text Archive text number 1243". These should be placed in an <idno> element within the appropriate division of the <publicationStmt> element. International Standard Serial Numbers (ISSN), International Standard Book Numbers (ISBN), and other internationally agreed upon standard numbers that uniquely identify an item, should be treated in the same way, rather than as specialized bibliographic notes.

Nevertheless, the <notesStmt> element may be used to record potentially significant details about the file and its features, e.g.:

- dates, when they are relevant to the content or condition of the computer file: e.g. "manual dated 1983," "Interview wave I: Apr. 1989; wave II: Jan. 1990"
- names of persons or bodies connected with the technical production, administration or consulting functions of the effort which produced the file, if these are not named in statements of responsibility in the title or edition statements of the file description: e.g. "Historical commentary provided by Mark Cohen"
- availability of the file in an additional medium or information not already recorded about the availability of documentation: e.g. "User manual is loose-leaf in eleven paginated sections"
- language of work and abstract: e.g. "Text in English with summaries in French and German"
- The unique name assigned to a serial by the International Serials Data System (ISDS)
- lists of related publications, either describing the source itself, or concerned with the creation or use of the machine-readable file, e.g. "Texts used in *Computation into Criticism* (Oxford, 1987)"

Each such item of information should be tagged using the general-purpose <note> element, which is described in section 6.8 ('Notes, Annotation, and Indexing') on p. 152. Groups of notes are contained within the <notesStmt> element, as in the following example:

```
<notesStmt>
  <note>Historical commentary provided by Mark Cohen.</note>
  <note>OCR scanning done at University of Toronto.</note>
</notesStmt>
```

The notes statement has the following formal definition:

```
<!-- 5.2.6:  The notes statement                              -->
<!ELEMENT notesStmt     - o  (note+)                          >
<!ATTLIST notesStmt          %a.global;                       >
<!-- The NOTE element is defined with the core tags.         -->

<!-- This fragment is used in sec. 5.2                       -->
```

5.2.7 The Source Description

The <sourceDesc> element is the seventh and final component of the <fileDesc> element. It is a mandatory element, and is used to record details of the source or sources from which a computer file is derived. This might be a printed text or manuscript, another computer file, an audio or video recording of some kind, or a combination of these. An electronic file may also have no source, if what is being catalogued is an original text created in electronic form.

<sourceDesc> supplies a bibliographic description of the copy text(s) from which an electronic text was derived or generated.

The <sourceDesc> element may contain a simple prose description, or, more usefully, a bibliographic citation of some kind specifying the provenance of the text. For written or printed sources, the source should be described in the same way as any other bibliographic citation, using one of the following elements:

<bibl> contains a loosely-structured bibliographic citation of which the sub-components may or may not be explicitly tagged.

<biblStruct> contains a structured bibliographic citation, in which only bibliographic subelements appear and in a specified order.

<biblFull> contains a fully-structured bibliographic citation, in which all components of the TEI file description are present.

<listBibl> contains a list of bibliographic citations of any kind.

These elements are described in more detail in section 6.10 ('Bibliographic Citations and References') on p. 162. When the header describes a transcription of spoken material, the <sourceDesc> element may also include the following special-purpose elements, intended for cases where an electronic text is derived from a spoken text rather than a written one:

<scriptStmt> contains a citation giving details of the script used for a spoken text.

<recordingStmt> describes a set of recordings used in transcription of a spoken text.

Full descriptions of these elements and their contents are given in section 5.2.9 ('Computer Files Composed of Transcribed Speech ') on p. 91. The <sourceDesc> element may contain a mixture of one or more of the above elements, as in the following examples:

```
<sourceDesc>
   <bibl>The first folio of Shakespeare, prepared by
   Charlton Hinman (The Norton Facsimile, 1968)</bibl>
</sourceDesc>

<sourceDesc>
   <p>No source: created in machine-readable form.</p>
</sourceDesc>

<sourceDesc>
   <biblStruct lang=FR>
   <monogr>
    <author>Eug&egrave;ne Sue</>
    <title>Martin, l'enfant trouv&eacute;</>
    <title type=sub>M&eacute;moires d'un valet de chambre</>
    <imprint>
         <pubPlace>Bruxelles et Leipzig</>
         <publisher>C. Muquardt</>
         <date>1846</>
    </imprint>
   </monogr></biblStruct>
</sourceDesc>
```

The source description itself has the following formal definition:

```
<!-- 5.2.7:  The source description                          -->
<!ELEMENT sourceDesc    - -  (p | bibl | biblFull | biblStruct
                              | listBibl | scriptStmt |
```

```
                              recordingStmt)+                    >
<!ATTLIST sourceDesc          %a.global;
                              %a.declarable;                     >
<!-- ... declarations from section 5.2.9                        -->
<!--     (Script statement and recording statement)             -->
<!--     go here ...                                            -->
<!-- This fragment is used in sec. 5.1.1                        -->
```

5.2.8 Computer Files Derived from Other Computer Files

If a machine-readable text (call it B) is based not on a printed source but upon another machine-readable text (call it A) which includes a TEI file header, then the source text of computer file B is another computer file, A. The four sections of A's file header will need to be incorporated into the new header for B in slightly differing ways, as listed below:

fileDesc A's file description should be copied into the <sourceDesc> section of B's file description, enclosed within a <biblFull> element (see section 6.10 ('Bibliographic Citations and References') on p. 162).

profileDesc A's <profileDesc> should be copied into B's, in principle unchanged.

encodingDesc A's coding practice may or (more likely) may not be the same as B's. Since the object of the coding description is to define the relationship between the current file and its source, in principle only changes in encoding practice between A and B need be documented in B. The relationship between A and its source(s) is then only recoverable from the original header of A. In practice it may be more convenient to create a new complete <encodingDesc> for B based on A's.

revisionDesc B is a new electronic file, and should therefore have a new revision description. If, however, it is felt useful to include some information from A's <revisionDesc>, for example dates of major updates or versions, such information must be clearly marked as relating to A rather than to B.

5.2.9 Computer Files Composed of Transcribed Speech

Where an electronic text is derived from a spoken text rather than a written one, it will usually be desirable to record additional information about the recording or broadcast which constitutes its source. Several additional elements are provided for this purpose within the source description element:

<scriptStmt> contains a citation giving details of the script used for a spoken text.

<recordingStmt> describes a set of recordings used in transcription of a spoken text.

<recording> details of an audio or video recording event used as the source of a spoken text, either directly or from a public broadcast. Attributes include:

- **type** the kind of recording. Legal values are:
  - ***audio*** audio recording
  - ***video*** audio and video recording
- **dur** the original duration of the recording.

<equipment> provides technical details of the equipment and media used for an audio or video recording used as the source for a spoken text.

<broadcast> describes a broadcast used as the source of a spoken text.

Note that detailed information about the participants or setting of an interview or other transcript of spoken language should be recorded in the appropriate division of the profile description, discussed in chapter 23 ('Language Corpora') on p. 537, rather than as part of the source description. The source description is used to hold information only about the source from which the transcribed speech was taken, for example, any script being read and any technical details of how the recording was produced. If the source was a previously-created transcript, it should be treated in the same way as any other source text.

The <scriptStmt> element should be used where it is known that one or more of the participants in a spoken text is speaking from a previously prepared script. The script itself should be documented in the same way as any other written text, using one of the three citation tags mentioned above. Utterances or groups of utterances may be linked to the script concerned by means of the decls attribute, described in section 23.3 ('Associating Contextual Information with a Text') on p. 550.

```
<sourceDesc>
  <scriptStmt id=CNN12>
  <bibl><author>CNN Network News</>
     <title>News headlines</>
     <date>12 Jun 1991</>
  </bibl>
  </scriptStmt>
  <!-- this script statement might be used to document
       the parts of a spoken transcript which included
       a news broadcast -->
  <!-- possibly other script statements or recording
       statements follow -->
</sourceDesc>
```

The <recordingStmt> is used to group together information relating to the recordings from which the spoken text was transcribed. The element may contain either a prose description or, more helpfully, one or more <recording> elements, each corresponding with a particular recording. The linkage between utterances or groups of utterances and the relevant recording statement is made by means of the decls attribute, described in section 23.3 ('Associating Contextual Information with a Text') on p. 550.

The <recording> element should be used to provide a description of how and by whom a recording was made. This information may be a prose description, within which such items as statements of responsibility, names, places and dates should be identified using the appropriate phrase level tags. The <recording> element takes two additional attributes, as indicated above: type is used to specify the kind of recording concerned and dur to specify its length.

In addition, descriptive information relating to the kind of recording equipment used should be specified using the <equipment> element. Where a recording is taken from a public broadcast, details of the broadcast should be given using the <broadcast> element described further below. Specialized collections may wish to add further sub-elements to these major components. Note however that this element should be used only for information relating to the recording process itself; information about the setting or participants (for example) is recorded elsewhere: see sections 23.2.3 ('The Setting Description') on p. 549 and 23.2.2 ('The Participants Description') on p. 545 below.

```
<recording type=video>
<p>U-matic recording made by college audio-visual
  department staff,available as PAL-standard VHS
  transfer or sound-only casssette
</recording>
```

```
<recording type=audio dur="30 mins">
<respStmt>
   <resp>Location recording by</resp>
   <name>Sound Services Ltd.</name>
</respStmt>
<equipment>
   <p>Multiple close microphones mixed down to stereo Digital
   Audio Tape, standard play, 44.1 KHz sampling frequency
</equipment>
<date>12 Jan 1987</date>
</recording>
```

When a recording has been made from a public broadcast, details of the broadcast itself should be supplied within the <recording> element, as a nested <broadcast> element. A broadcast is closely analogous to a publication and the <broadcast> element should therefore

contain one or the other of the bibliographic citation elements <bibl>, <biblStruct> or <biblFull>. The broadcasting agency responsible for a broadcast is regarded as its author, while other participants (for example interviewers, interviewees, directors, producers, etc.) should be specified using the <respStmt> or <editor> element with an appropriate <resp> (see further section 6.10 ('Bibliographic Citations and References') on p. 162).

```
<recording type=audio dur="10 mins">
  <equipment><p>Recorded from FM Radio to digital tape</>
  <broadcast>
     <bibl><title>Interview on foreign policy</>
              <author>BBC Radio 5</>
       <respStmt><resp>interviewer</><name>Robin Day</></>
       <respStmt><resp>interviewee</><name>Margaret Thatcher</></>
       <series><title>The World Tonight</></>
       <note>First broadcast on <date>27 Nov 1989</></note>
    </bibl>
  </broadcast>
</recording>
```

When a broadcast contains several distinct recordings (for example a compilation), additional <recording> elements may be further nested within the <broadcast> element.

```
    <recording dur=100>
       <broadcast>
         <!-- details of broadcast -->
         <recording>
            <!-- details of broadcast recording -->
         </recording>
       </broadcast>
    </recording>
```

Formal definitions for the elements discussed in this section are as follows:

```
<!-- 5.2.9:  Script statement and recording statement         -->
<!ELEMENT scriptStmt    - -  (p+ | bibl | biblFull |
                              biblStruct)                      >
<!ATTLIST scriptStmt         %a.global;
                              %a.declarable;                   >
<!ELEMENT recordingStmt - -  (p+ | recording+ )                >
<!ATTLIST recordingStmt      %a.global;                        >
<!ELEMENT recording     - -  (p+ | (respStmt | equipment |
                              broadcast | date)*)              >
<!ATTLIST recording          %a.global;
                              %a.declarable;
           type               (audio | video)     audio
           dur                CDATA               #IMPLIED     >
<!ELEMENT equipment     - o  (p+)                              >
<!ATTLIST equipment          %a.global;
                              %a.declarable;                   >
<!ELEMENT broadcast     - -  (p+ | bibl | biblStruct | biblFull
                              | recording)                     >
<!ATTLIST broadcast          %a.global;
                              %a.declarable;                   >
<!-- This fragment is used in sec. 5.2.7                      -->
```

This concludes the discussion of the <fileDesc> element and its contents.

5.3 The Encoding Description

The <encodingDesc> element is the second major subdivision of the TEI header. It specifies the methods and editorial principles which governed the transcription or encoding of the text

in hand and may also include sets of coded definitions used by other components of the header. Though not formally required, its use is highly recommended.

<encodingDesc> documents the relationship between an electronic text and the source or sources from which it was derived.

The content of the encoding description may be a prose description, or it may contain elements from the following list, in the order given:

<projectDesc> describes in detail the aim or purpose for which an electronic file was encoded, together with any other relevant information concerning the process by which it was assembled or collected.

<samplingDecl> contains a prose description of the rationale and methods used in sampling texts in the creation of a corpus or collection.

<editorialDecl> provides details of editorial principles and practices applied during the encoding of a text.

<tagsDecl> provides detailed information about the tagging applied to an SGML document.

<refsDecl> specifies how canonical references are constructed for this text.

<classDecl> contains one or more taxonomies defining any classificatory codes used elsewhere in the text.

<fsdDecl> identifies the feature system declaration which contains definitions for a particular type of feature structure.

<metDecl> documents the notation employed to represent a metrical pattern when this is specified as the value of a met, real, or rhyme attribute on any structural element of a metrical text (e.g. <lg>, <l>, or <seg>).

<variantEncoding> declares the method used to encode text-critical variants.

Each of these elements is further described and formally defined in the appropriate section below. The encoding description itself is defined as follows:

```
<!-- 5.3:  The encoding description                          -->
<!ELEMENT encodingDesc  - -  (projectDesc*, samplingDecl*,
                              editorialDecl*, tagsDecl?,
                              refsDecl*, classDecl*, metDecl*,
                              fsdDecl*, p*)                    >
<!ATTLIST encodingDesc       %a.global;                        >
<!-- ... declarations from section 5.3.1                      -->
<!--     (The project description)                            -->
<!--     go here ...                                          -->
<!-- ... declarations from section 5.3.2                      -->
<!--     (The sampling declaration)                           -->
<!--     go here ...                                          -->
<!-- ... declarations from section 5.3.3                      -->
<!--     (The editorial practices declaration)                -->
<!--     go here ...                                          -->
<!-- ... declarations from section 5.3.4                      -->
<!--     (Tag usage and rendition declarations)               -->
<!--     go here ...                                          -->
<!-- ... declarations from section 5.3.5.3                    -->
<!--     (The reference scheme declaration)                   -->
<!--     go here ...                                          -->
<!-- ... declarations from section 5.3.6                      -->
<!--     (The classification declaration)                     -->
<!--     go here ...                                          -->
<!-- ... declarations from section 5.3.7                      -->
<!--     (The FSD declaration)                                -->
<!--     go here ...                                          -->
<!-- ... declarations from section 5.3.8                      -->
<!--     (Metrical Notation Declaration)                      -->
<!--     go here ...                                          -->
<!-- ... declarations from section 5.3.9                      -->
<!--     (Variant-Encoding Declaration)                       -->
```

```
<!--     go here ...                                          -->
<!-- This fragment is used in sec. 5.1.1                      -->
```

5.3.1 The Project Description

The <projectDesc> element is the first of the nine optional subdivisions of the <encodingDesc> element. It may be used to describe, in prose, the purpose for which the electronic file was encoded, together with any other relevant information concerning the process by which it was assembled or collected. This is of particular importance for corpora or miscellaneous collections, but may be of use for any text, for example to explain why one kind of encoding practice has been followed rather than another.

<projectDesc> describes in detail the aim or purpose for which an electronic file was encoded, together with any other relevant information concerning the process by which it was assembled or collected.

For example:

```
<encodingDesc>
<projectDesc><p>Texts collected for use in the
   Claremont Shakespeare Clinic, June 1990.
</encodingDesc>
```

This element has the following formal declaration:

```
<!-- 5.3.1:  The project description                          -->
<!ELEMENT projectDesc   - o  (p+)                              >
<!ATTLIST projectDesc        %a.global;
                             %a.declarable;                    >
<!-- This fragment is used in sec. 5.3                        -->
```

5.3.2 The Sampling Declaration

The <samplingDecl> element is the second of the nine optional subdivisions of the <encodingDesc> element. It contains a prose description of the rationale and methods used in sampling texts, for example to create a representative corpus.

<samplingDecl> contains a prose description of the rationale and methods used in sampling texts in the creation of a corpus or collection.

It should include information about such matters as

- the size of individual samples
- the method or methods by which they were selected
- the underlying population being sampled
- the object of the sampling procedure used

but is not restricted to these.

```
<samplingDecl>
  <p>Samples of 2000 words taken from the beginning of the text.
</samplingDecl>
```

It may also include a simple description of any parts of the source text included or excluded.

```
<samplingDecl>
  <p>Text of stories only has been transcribed.
     Pull quotes, captions, and advertisements have been
     silently omitted.  Any mathematical expressions
     requiring symbols not present in the ISOnum or
     ISOpub entity sets have been omitted, and their
     place marked with a GAP element.
</samplingDecl>
```

A sampling declaration which applies to more than one text or division of a text need not be repeated in the header of each such text. Instead, the decls attribute of each text (or subdivision of the text) to which the sampling declaration applies may be used to supply a cross reference to it, as further described in section 23.3 ('Associating Contextual Information with a Text') on p. 550. This element has the following formal declaration:

```
<!-- 5.3.2:  The sampling declaration                          -->
<!ELEMENT samplingDecl  - o  (p+)                               >
<!ATTLIST samplingDecl       %a.global;
                             %a.declarable;                     >
<!-- This fragment is used in sec. 5.3                         -->
```

5.3.3 The Editorial Practices Declaration

The <editorialDecl> element is the third of the nine optional subdivisions of the <encodingDesc> element. It is used to provide details of the editorial practices applied during the encoding of a text.

<editorialDecl> provides details of editorial principles and practices applied during the encoding of a text.

It may contain a prose description only, or one or more of the following specialized elements:

<correction> states how and under what circumstances corrections have been made in the text. Attributes include:

status indicates the degree of correction applied to the text. Legal values are:

high the text has been thoroughly checked and proofread.

medium the text has been checked at least once.

low the text has not been checked.

unknown the correction status of the text is unknown.

method indicates the method adopted to indicate corrections within the text. Legal values are:

silent corrections have been made silently

tags corrections have been represented using editorial tags

<normalization> indicates the extent of normalization or regularization of the original source carried out in converting it to electronic form. Attributes include:

source indicates the authority for any normalization carried out.

method indicates the method adopted to indicate normalizations within the text. Sample values include:

silent normalization made silently

tags normalization represented using editorial tags

<quotation> specifies editorial practice adopted with respect to quotation marks in the original. Attributes include:

marks indicates whether or not quotation marks have been retained as content within the text. Sample values include:

none no quotation marks have been retained

some some quotation marks have been retained

all all quotation marks have been retained

form specifies how quotation marks are indicated within the text. Sample values include:

data quotation marks are retained as data.

rend the rendition attribute is consistently used to indicate the form of quotation marks.

std use of quotation marks has been standardized.

nonstd quotation marks are represented inconsistently.

unknown use of quotation marks is unknown.

<hyphenation> summarizes the way in which hyphenation in a source text has been treated in an encoded version of it. Attributes include:

eol indicates whether or not end-of-line hyphenation has been retained in a text. Sample values include:

all all end-of-line hyphenation has been retained, even though the lineation of the original may not have been.

some end-of-line hyphenation has been retained in some cases.

hard all 'soft' end-of-line hyphenation has been removed: any remaining hyphenation represents 'hard hyphens'.

none all end-of-line hyphenation has been removed: any remaining hyphenation occurred within the line.

<segmentation> describes the principles according to which the text has been segmented, for example into sentences, tone-units, graphemic strata, etc.

<stdVals> specifies the format used when standardized date or number values are supplied.

<interpretation> describes the scope of any analytic or interpretive information added to the text in addition to the transcription.

Some of these elements carry attributes to support automated processing of certain well-defined editorial decisions; all of them contain a prose description of the editorial principles adopted with respect to the particular feature concerned. Examples of the kinds of questions which these descriptions are intended to answer are listed below, in the same order as the list above.

<correction> Was the text corrected during or after data capture? If so, were corrections made silently or are they marked using the tags described in section 6.5 ('Simple Editorial Changes') on p. 140? What principles have been adopted with respect to omissions, truncations, dubious corrections, alternate readings, false starts, repetitions, etc.?

<normalization> Was the text normalized, for example by regularizing any non-standard spellings, dialect forms, etc.? If so, were normalizations performed silently or are they marked using the tags described in section 6.5 ('Simple Editorial Changes') on p. 140? What authority was used for the regularization? Also, what principles were used when normalizing dates or numbers to provide the standard values for the value attribute described in sections 6.4.3 ('Numbers and Measures') on p. 135 and 6.4.4 ('Dates and times') on p. 137 and what format used for them?

<quotation> How were quotation marks processed? Are apostrophes and quotation marks distinguished? How? Are quotation marks retained as content in the text or replaced by markup? Was the rendition attribute used to record the specific appearance of any quotation marks removed from the text? Are there any special conventions regarding for example the use of single or double quotation marks when nested? Is the file consistent in its practice or has this not been checked?

<hyphenation> Does the encoding distinguish "soft" and "hard" hyphens? What principle has been adopted with respect to end-of-line hyphenation where source lineation has not been retained? Have soft hyphens been silently removed, and if so what is the effect on lineation and pagination?

<segmentation> How is the text segmented? If <s> or <seg> segmentation units have been used to divide up the text for analysis, how are they marked and how was the segmentation arrived at?

<stdVals> What standardization methods underly any standardized values supplied for numeric values or dates? If the value attribute described in section 6.4.4 ('Dates and times') on p. 137 has been used, in what format are its values presented?

<interpretation> Has any analytic or "interpretive" information been provided — that is, information which is felt to be non-obvious, contentious, or subject to disagreement? If so, how was it generated? How was it encoded? If feature-structure analysis has been used, are <fsdDecl> elements (section 5.3.7 ('The Feature System Declaration') on p. 105) present?

Any information about the editorial principles applied not falling under one of the above headings should be recorded in a distinct list of items. Experience shows that a full record should be kept of decisions relating to editorial principles and encoding practice, both for future users of the text and for the project which produced the text in the first instance.

A simple example follows:

```
<editorialDecl id=E2>
<interpretation>
  <p>The part of speech analysis applied throughout section 4 was
  added by hand and has not been checked.
```

```
<correction><p>Errors in transcription controlled by using the
    WordPerfect spelling checker.
<normalization source=W9>
  <p>All words converted to Modern American spelling using
    Websters 9th Collegiate dictionary.
<quotation marks=all form=std>
  <p>All opening quotation marks converted to &odq; all closing
    quotation marks converted to &cdq;.
</editorialDecl>
```

These elements are formally defined as follows:

```
<!-- 5.3.3:  The editorial practices declaration              -->
<!ELEMENT editorialDecl - o  ( p+ | ((correction |
                             normalization | quotation |
                             hyphenation | interpretation |
                             segmentation | stdVals)+, p*))     >
<!ATTLIST editorialDecl      %a.global;
                             %a.declarable;                     >
<!ELEMENT correction    - o  (p+)                               >
<!ATTLIST correction         %a.global;
                             %a.declarable;
          status             (high | medium | low | unknown)
                                                    unknown
          method             (silent | tags)      silent        >
<!ELEMENT normalization - o  (p+)                               >
<!ATTLIST normalization      %a.global;
                             %a.declarable;
          source             CDATA                #IMPLIED
          method             (silent | tags)      silent        >
<!ELEMENT quotation     - o  (p+)                               >
<!ATTLIST quotation          %a.global;
                             %a.declarable;
          marks              (none | some | all) all
          form               (data | rend | std | nonstd |
                             unknown)             unknown       >
<!ELEMENT hyphenation   - o  (p+)                               >
<!ATTLIST hyphenation        %a.global;
                             %a.declarable;
          eol                (all | some | none) some          >
<!ELEMENT segmentation  - o  (p+)                               >
<!ATTLIST segmentation       %a.global;
                             %a.declarable;                     >
<!ELEMENT stdVals       - o  (p+)                               >
<!ATTLIST stdVals            %a.global;
                             %a.declarable;                     >
<!ELEMENT interpretation
                        - o  (p+)                               >
<!ATTLIST interpretation     %a.global;
                             %a.declarable;                     >
<!-- This fragment is used in sec. 5.3                          -->
```

An editorial practices declaration which applies to more than one text or division of a text need not be repeated in the header of each such text. Instead, the decls attribute of each text (or subdivision of the text) to which it applies may be used to supply a cross reference to it, as further described in section 23.3 (‘Associating Contextual Information with a Text’) on p. 550.

5.3.4 The Tagging Declaration

The <tagsDecl> element is the fourth of the nine optional subdivisions of the <encodingDesc> element. It is used to record the following information about the tagging used within a particular text:

- how often particular SGML elements appear within the text, so that a recipient can validate the integrity of a text during interchange.
- any comment relating to the usage of particular elements not specified elsewhere in the header.
- a definition for the default rendition applying to all instances of an element, unless otherwise stated by the global rend attribute.

This information is conveyed by the following elements:

<rendition> supplies information about the intended rendition of one or more elements.

<tagUsage> supplies information about the usage of a specific element within a <text>. Attributes include:

occurs specifies the number of occurrences of this element within the text.

ident specifies the number of occurrences of this element within the text which bear a distinct value for the global id attribute.

render specifies the identifier of a <rendition> element which defines how this element is to be rendered.

The <tagsDecl> element consists of an optional sequence of <rendition> elements, each of which must bear a unique identifier, followed by a sequence of <tagUsage> elements, one for each distinct element occurring within the outermost <text> element of a TEI document.

The <rendition> element defined in this version of the TEI Guidelines is a preliminary proposal only, intended to provide a hook for more detailed specifications of default rendition in later versions.

The present proposal allows the encoder to enter an informal description of a rendition, or style, as running prose only. This rendition will be assumed to apply, by default, to all occurrences of an element which names its identifier as the value of the render attribute of the appropriate <tagUsage> element. For element occurrences to which this default rendition does not apply, the encoder should specify an explicit description using the global rend attribute on the elements concerned.

For example, the following schematic shows how an encoder might specify that <p> elements are by default to be rendered using one set of specifications identified as style1, while <hi> elements are to use a different set, identified as style2:

```
<tagsDecl>
 <rendition id=style1>
   <!-- description of one default rendition here -->
 <rendition id=style2>
   <!-- description of another default rendition here -->
 <tagUsage gi=p render=style1>
 <tagUsage gi=hi render=style2>
</tagsDecl>
```

No detailed proposals for the content of the <rendition> element have as yet been formulated. It is probable that subsequent work will incorporate specifications derived from, or compatible with, the properties currently being standardized as part of the draft Document Style and Semantics Specification Language (ISO DIS 10179).

A <tagsDecl> need not specify any <rendition> element. If present, it must however contain exactly one occurrence of a <tagUsage> element for each distinct element marked within the outermost <text> element associated with the <teiHeader> in which it appears.[5] The <tagUsage> element is used to supply a count of the number of occurrences of this element within the text, which is given as the value of its occurs attribute. It may also be used to hold any additional usage information, which is supplied as running prose within the element itself.

For example:

```
<tagUsage gi=hi occurs=28>
    Used only to mark English words italicised in the copy text.
</tagUsage>
```

This indicates that the <hi> element appears a total of 28 times in the <text> element in question, and that the encoder has used it to mark italicised English phrases only.

[5] This implies that the <tagsDecl> will normally only appear in the header of individual texts in a <teiCorpus>.

The ident attribute may optionally be used to specify how many of the occurrences of the element in question bear a value for the global id attribute, as in the following example:

```
<tagUsage gi=pb occurs=321 ident=321>
    Marks page breaks in the York (1734) edition only
</tagUsage>
```

This indicates that the <pb> element occurs 321 times, on each of which an identifier is provided.

The content of the <tagUsage> element is not susceptible of automatic processing. It should not therefore be used to hold information for which provision is already made by other components of the encoding description. A TEI conformant document is not required to contain a <tagsDecl> element, but if one is present, it must contain <tagUsage> elements for each distinct element marked in the associated text, and the counts specified by their usage elements must correspond with the number of such elements present in the document, as identified by some conforming SGML processor.

```
<!-- 5.3.4:  Tag usage and rendition declarations             -->
<!ELEMENT tagsDecl      - O  (rendition*, tagUsage*)           >
<!ATTLIST tagsDecl           %a.global;                        >
<!ELEMENT tagUsage      - O  (%paraContent)                    >
<!ATTLIST tagUsage           %a.global;
          gi                 NAME                #REQUIRED
          occurs             NUMBER              #IMPLIED
          ident              NUMBER              #IMPLIED
          render             IDREF               #IMPLIED      >
<!ELEMENT rendition     - O  (%paraContent)                    >
<!ATTLIST rendition          %a.global;                        >
<!-- This fragment is used in sec. 5.3                        -->
```

5.3.5 The Reference System Declaration

The <refsDecl> element is the fifth of the nine optional subdivisions of the <encodingDesc> element. It is used to document the way in which any standard referencing scheme built into the encoding works, either as a series of prose paragraphs or by using the following specialized elements:

<refsDecl> specifies how canonical references are constructed for this text. Attributes include:
  doctype identifies the *document type* within which this reference declaration is used.

<step> specifies one component of a canonical reference defined by the "stepwise" method.

<state> specifies one component of a canonical reference defined by the "milestone" method.

Note that not all possible referencing schemes are equally easily supported by current software systems. A choice must be made between the convenience of the encoder and the likely efficiency of the particular software applications envisaged, in this context as in many others. For a more detailed discussion of referencing systems supported by these Guidelines, see section 6.9 ('Reference Systems') on p. 155 below.

A referencing scheme may be described in one of three ways using this element:

- as a prose description
- as a series of *steps* expressed in the TEI extended pointer notation (documented in section 14.2 ('Extended Pointers') on p. 340)
- as a concatenation of sequentially organized *milestones*

Each method is described in more detail below. Only one method can be used within a single <refsDecl> element.

More than one <refsDecl> element can be included in the header if more than one canonical reference scheme is to be used in the same document, but the current proposals do not check for mutual inconsistency. A reference declaration can only describe the referencing system applicable to a single document type; if therefore concurrent document types are in use (as discussed in section 6.9 ('Reference Systems') on p. 155), a <refsDecl> element must be supplied for each; the doctype attribute should be used to specify the document type to which the declaration relates.

5.3.5.1 Prose Method

The referencing scheme may be specified within the <refsDecl> by a simple prose description. Such a description should indicate which elements carry identifying information, and whether this information is represented as attribute values or as content. Any special rules about how the information is to be interpreted when reading or generating a reference string should also be specified here. Such a prose description cannot be processed automatically, and this method of specifying the structure of a canonical reference system is therefore not recommended for automatic processing.

For example:

```
<refsDecl>
<p>The N attribute of each text in this corpus carries
a unique identifying code for the whole text. The title
of the text is held as the content of the first HEAD
element within each text. The N attribute on each DIV1
and DIV2 contains the canonical reference for
each such division, in the form 'XX.yyy', where XX
is the book number in Roman numerals, and yyy the section
number in arabic. Line breaks are marked by empty LINEBREAK
tags, each of which includes the through line number
in Casaubon's edition as the value of its N attribute.
<p>The through line number and the text identifier
uniquely identify any line. A canonical reference
may be made up by concatenating the the N values from the
text, div1 or div2 and calculating the line number within
each part.
</refsDecl>
```

5.3.5.2 Stepwise Method

This method defines each reference as a series of *steps*, each of which corresponds to a single pair of expressions in the TEI extended pointer notation (for which see section 14.2 ('Extended Pointers') on p. 340). Often, but not always, each step will also correspond to one portion of the canonical reference itself; in many common forms of canonical reference, each step will narrow the scope within which the next step can be taken. The <refsDecl> element must specify the steps, delimiters and lengths to be used by an application program, both when constructing references for a given location and when interpreting canonical references within a given document hierarchy. It does so by supplying one or more <step> elements, each of which identifies the type of "reference unit" handled by the step and uses a pair of extended-pointer expressions to indicate the starting and ending pointers of the portion of the document which corresponds to a given portion of the reference string. The element may also give either a delimiter or a length for use in breaking the corresponding reference string up into units.

<step> specifies one component of a canonical reference defined by the "stepwise" method. Attributes include:

- **refunit** names the unit (book, chapter, canto, verse, ...) identified by this step in a canonical reference.
- **from** specifies the starting point of the area referred to by this step in the canonical reference.
- **to** specifies the ending point of the area referred to by this step in the canonical reference.
- **delim** supplies a delimiting string following the reference component.
- **length** specifies the fixed length of the reference component.

For example, the reference "Matthew 5:29" might be constructed by stepping down the tree to find an element labelled as the "Matthew" node, then within that to the "5" node, and finally, within that, to the "29" node. The following declarations would be required; the special values %1, %2, and %3 refer here to the strings 'Matthew', '5', and '29', respectively.

```
<refsDecl>
    <step refunit='book' delim=' '
          from='DESCENDANT (1 DIV1 N %1)'>
```

```
    <step refunit='chapter' delim=':'
          from='DESCENDANT (1 DIV2 N %2)'>
    <step refunit='verse'
          from='DESCENDANT (1 DIV3 N %3)'>
</refsDecl>
```

As this example also shows, the steps of such a reference are typically separated by fixed character sequences, called *delimiters*. In this example, the delimiters are a space (following "Matthew") and a colon (following the chapter number). A processor for canonical references would use the delimiters specified by the delim attributes to break the reference string up into pieces; the pieces would then be used to interpret the %1, etc., in the extended pointer expressions of the from and to attributes.

An alternative to the use of delimiters is to specify a fixed length for each step of the reference: for example, the same reference might be given as "MAT05029", assuming a fixed length of 3 for the first step, 2 for the second and 3 for the third.

```
<refsDecl>
    <step length=3
          from='DESCENDANT (1 DIV1 N %1)'>
    <step length=2
          from='DESCENDANT (1 DIV2 N %2)'>
    <step length=3
          from='DESCENDANT (1 DIV3 N %3)'>
</refsDecl>
```

The order in which the <step> elements are supplied corresponds here with the order of elements within the reference, with the largest (that is, the one nearest the top of the document hierarchy) item first and the smallest last.

For a description of the processing required when a canonical reference defined by <step> elements is to be recognized, and examples of its use, see chapter 32 ('Algorithm for Recognizing Canonical References') on p. 641.

5.3.5.3 Milestone Method

This method is appropriate when only "milestone" tags (see section 6.9.3 ('Milestone Tags') on p. 158) are available to provide the required referencing information. It does not provide any abilities which cannot be mimicked by the stepwise referencing method discussed in the previous section, but in the cases where it applies, it provides a somewhat simpler notation.

A reference based on milestone tags concatenates the values specified by one or more such tags. Since each tag marks the point at which a value changes, it may be regarded as specifying the *state* of a variable. A reference declaration using this method therefore specifies the individual components of the canonical reference as a sequence of <state> elements:

<state> specifies one component of a canonical reference defined by the "milestone" method. Attributes include:

ed indicates which edition or version the milestone applies to.

unit indicates what kind of section is changing at this milestone. Suggested values include:

page page breaks in the reference edition.

column column breaks.

line line breaks.

book any units termed "book","liber", etc.

poem individual poems in a collection.

canto cantos or other major sections of a poem.

stanza stanzas within a poem, book, or canto.

act acts within a play.

scene scenes within a play or act.

section sections of any kind.

absent passages not present in the reference edition.

delim supplies a delimiting string following the reference component.

length specifies the fixed length of the reference component.

For example, the reference "Matthew 12:34" might be thought of as representing the state of three variables: the "book" variable is in state "Matthew"; the *chapter* variable is in state "12", and the *verse* variable is in state "34". If milestone tagging has been used, there should be a tag marking the point in the text at which each of the above "variables" changes its state.[6] To find "Matthew 12:34" therefore an application must scan left to right through the text, monitoring changes in the state of each of these three variables as it does so. When all three are simultaneously in the required state, the desired point will have been reached. There may of course be several such points.

The delim and length attributes are used to specify components of a canonical reference using this method in exactly the same way as for the stepwise method described in the preceding section. The other attributes are used to determine which instances of <milestone> tags in the text are to be checked for state-changes. A state-change is signalled whenever a new <milestone> tag is found with unit and, optionally, ed attributes identical to those of the <state> element in question. The value for the new state may be given explicitly by the n attribute on the <milestone> element, or it may be implied, if the n attribute is not specified.

For example, for canonical references in the form 'xx.yyy' where the 'xx' represents the page number in the first edition, and 'yyy' the line number within this page, a reference system declaration such as the following would be appropriate:

```
<refsDecl>
  <state ed='first' unit='page' delim='.' length=2>
  <state ed='first' unit='line' length=4>
</refsDecl>
```

This implies that milestone tags of the form

```
<milestone ed='first' unit=page n='II'>
<milestone ed='first' unit=line>
```

will be found throughout the text, marking the positions at which page and line numbers change. Note that no value has been specified for the n attribute on the second milestone tag above; this implies that its value at each state change is monotonically increased. For more detail on the use of milestone tags, see section 6.9.3 ('Milestone Tags') on p. 158.

The milestone referencing scheme, though conceptually simple, is not and cannot be supported by an SGML parser. Its use places a correspondingly greater burden of verification and accuracy on the encoder.

The elements discussed in this section are formally defined as follows:

```
<!-- 5.3.5.3:  The reference scheme declaration                -->
<!ELEMENT refsDecl      - o  (p+ | step+ | state+)              >
<!ATTLIST refsDecl           %a.global;
          doctype            NAME                TEI.2        >
<!ELEMENT step          - o  EMPTY                               >
<!ATTLIST step               %a.global;
          refunit            CDATA               #IMPLIED
          length             NUMBER              #IMPLIED
          delim              CDATA               #IMPLIED
          from               %extPtr             #REQUIRED
          to                 %extPtr             'DITTO'      >
<!ELEMENT state         - o  EMPTY                               >
<!ATTLIST state              %a.global;
          ed                 CDATA               #IMPLIED
          unit               CDATA               #REQUIRED
          length             NUMBER              #IMPLIED
          delim              CDATA               #IMPLIED     >
<!-- This fragment is used in sec. 5.3                          -->
```

A reference system declaration which applies to more than one text or division of a text need not be repeated in the header of each such text. Instead, the decls attribute of each text (or

[6] On the milestone tag itself, what are here referred to as "variables" are identified by the combination of the ed and unit attributes.

subdivision of the text) to which the declaration applies may be used to supply a cross reference to it, as further described in section 23.3 ('Associating Contextual Information with a Text') on p. 550.

5.3.6 The Classification Declaration

The <classDecl> element is the sixth of the nine optional subdivisions of the <encodingDesc> element. It is used to group together definitions or sources for any descriptive classification schemes used by other parts of the header. Each such scheme is represented by a <taxonomy> element, which may contain either a simple bibliographic citation, or a definition of the descriptive typology concerned; the following elements are used in defining a descriptive classification scheme:

<classDecl> contains one or more taxonomies defining any classificatory codes used elsewhere in the text.

<taxonomy> defines a typology used to classify texts either implicitly, by means of a bibliographic citation, or explicitly by a structured taxonomy.

<category> contains an individual descriptive category, possibly nested within a superordinate category, within a user-defined taxonomy.

<catDesc> describes some category within a taxonomy or text typology, either in the form of a brief prose description
or in terms of the situational parameters used by the TEI
formal <textDesc>.

The <taxonomy> element has two slightly different, but related, functions. For well-recognized and documented public classification schemes, such as Dewey or other published descriptive thesauri, it contains simply a bibliographic citation indicating where a full description of a particular taxonomy may be found.

```
<taxonomy id=DDC12>
  <bibl><title>Dewey Decimal Classification</title>
    <edition>Abridged Edition 12</edition>
    <!-- etc. -->
  </bibl>
</taxonomy>
```

For less easily accessible schemes, the <taxonomy> element contains a description of the taxonomy itself as well as an optional bibliographic citation. The description consists of a number of <category> elements, each defining a single category within the given typology. The category is defined by the contents of a nested <catDesc> element, which may contain either a phrase describing the category, or a <textDesc> element defining it in terms of the situational parameters discussed in section 23.2.1 ('The Text Description') on p. 541. If the category is subdivided, each subdivision is represented by a nested <category> element, having the same structure. Categories may be nested to an arbitrary depth in order to reflect the hierarchical structure of the taxonomy. Each <category> element bears a unique id attribute, which is used as the target for <catRef> elements referring to it.

```
<taxonomy id=B>
   <bibl>Brown Corpus</bibl>
   <category id=B.A><catDesc>Press Reportage
      <category id=B.A1><catDesc>Daily</category>
      <category id=B.A2><catDesc>Sunday</category>
      <category id=B.A3><catDesc>National</category>
      <category id=B.A4><catDesc>Provincial</category>
      <category id=B.A5><catDesc>Political</category>
      <category id=B.A6><catDesc>Sports</category>
     ...
   </category>
   <category id=B.D><catDesc>Religion
      <category id=B.D1><catDesc>Books</category>
```

```
      <category id=B.D2><catDesc>Periodicals and tracts</category>
    </category>
   ...
</taxonomy>
```

Linkage between a particular text and a category within such a taxonomy is made by means of the <catRef> element within the <textClass> element, as described in section 5.4.3 ('The Text Classification') on p. 111. Where the taxonomy permits of classification along more than one dimension, more than one category will be referenced by a particular <catRef>, as in the following example, which identifies a text with the sub-categories "Daily", "National" and "Political", within the category "Press Reportage" as defined above.

```
<catRef target="B.A1 B.A3 B.A5">
```

The elements discussed in this section are defined as follows:

```
<!-- 5.3.6:  The classification declaration                    -->
<!ELEMENT classDecl     - -  (taxonomy+)                         >
<!ATTLIST classDecl          %a.global;                          >
<!ELEMENT taxonomy      - -  (category+ | ((bibl | biblStruct |
                             biblFull), category*))              >
<!ATTLIST taxonomy           %a.global;                          >
<!ELEMENT category      - -  (catDesc, category*)                >
<!ATTLIST category           %a.global;                          >
<!ELEMENT catDesc       - o  (%phrase.seq; | textDesc)           >
<!ATTLIST catDesc            %a.global;                          >
<!-- This fragment is used in sec. 5.3                          -->
```

5.3.7 The Feature System Declaration

The <fsdDecl> element is the seventh of the nine optional subdivisions of the <encodingDesc> element. It is used to associate a *feature system declaration* (as defined in chapter 26 ('Feature System Declaration') on p. 589) with any analytic *feature structures* (as defined in chapter 16 ('Feature Structures') on p. 397) present in the text documented by this header.

It has the following description and attributes:

<fsdDecl> identifies the feature system declaration which contains definitions for a particular type of feature structure. Attributes include:

- **type** identifies the type of feature structure documented in the FSD; this will be the value of the type attribute on at least one feature structure.
- **fsd** specifies the external entity containing the feature system declaration; an entity declaration in the document's DTD subset must associate the entity name with a file on the system.

Note that one <fsdDecl> element must be specified for each distinct type of feature structure used in the markup. The fsd element supplies the name of an external entity containing the actual declaration for that type of feature structure. This entity will typically be defined in the document type subset for the document, as in the following example:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
   <!-- declare an external entity for the wsd -->
   <!ENTITY myFeatures  SYSTEM 'myfeat.fsd' SUBDOC >
]>
<tei.2>
...
</tei.2>
```

This associates the name *myFeatures* with an external SUBDOC entity[7] held in this example within a system file called *myfeat.wsd*. This entity name may then be specified within a <fsdDecl> element in the header to inform a processor of the location of the feature system declaration corresponding to a given type of feature structure used within the text, as follows:

[7] The SUBDOC keyword indicates to an SGML parser that this entity contains SGML data which must be parsed using some other DTD than the current one: in this case, the DTD defined in chapter 26 ('Feature System Declaration') on p. 589 rather than the view of the TEI DTD defined for the document itself.

```
<teiHeader>
   <fileDesc> ... </fileDesc>
   <encodingDesc>
        <!-- ... -->
        <fsdDecl type=myA1 fsd=myFeatures>
        <fsdDecl type=myA2 fsd=myFeatures>
        <!-- ... -->
   </encodingDesc>
```

This header would be attached to a text in which feature structures of types myA1 and myA2 are used.

Further details and examples of the use of feature structure analyses and feature system declarations are provided in chapters 16 ('Feature Structures') on p. 397 and 26 ('Feature System Declaration') on p. 589 respectively.

The <fsdDecl> element is declared as follows:

```
<!-- 5.3.7:  The FSD declaration                                  -->
<!ELEMENT fsdDecl       - O  EMPTY                                 >
<!ATTLIST fsdDecl            %a.global;
          type               CDATA                #REQUIRED
          fsd                ENTITY               #REQUIRED      >
<!-- This fragment is used in sec. 5.3                            -->
```

5.3.8 The Metrical Declaration Element

The <metDecl> element is the eighth of the nine optional subdivisions of the <encodingDesc> element. It is used to document any metrical notation scheme used in the text, as further discussed in section 9.4 ('Rhyme and Metrical Analysis') on p. 221. It consists either of a prose description or a series of <symbol> elements.

<metDecl> documents the notation employed to represent a metrical pattern when this is specified as the value of a met, real, or rhyme attribute on any structural element of a metrical text (e.g. <lg>, <l>, or <seg>). Attributes include:

pattern specifies a regular expression defining any value that is legal for this notation.

<symbol> documents the intended significance of a particular character or character sequence within a metrical notation, either explicitly or in terms of other <symbol> elements in the same <metNotation>. Attributes include:

value specifies the character or character sequence being documented.

terminal specifies whether the symbol is defined in terms of other symbols (terminal=N) or in prose (terminal=Y). Sample values include:

Y the element contains a prose definition of its meaning.

N the element contains a definition of its meaning given using symbols defined elsewhere in the same <metNotation> element.

As with other components of the header, metrical notation may be specified either formally or informally. In a formal specification, every symbol used in the metrical notation must be documented by a corresponding <symbol> element; in an informal one, only a brief prose description of the way in which the notation is used need be given. In either case, the optional pattern attribute may be used to supply a regular expression which a processor can use to validate expressions in the intended notation. The following constraints apply:

- if pattern is supplied, any notation used which does not conform to it should be regarded as invalid
- if any <symbol> is defined, then any notation using undefined symbols should be regarded as invalid
- if both pattern and symbol are defined, then every symbol appearing explicitly within pattern must be defined
- symbols may be defined within a <metDecl> element which are not matched by pattern

As a simple example, consider the case of the notation in which metrical prominence, foot and line boundaries are all to be encoded. Legal specifications in this notation may be written for

any sequence of metrically prominent or non-prominent features, optionally separated by foot or metrical line boundaries at arbitrary points. Assuming that the symbol '1' is used for metrical prominence, '0' for non-prominence, '|' for foot boundary and '/' for line boundary, then the following declaration achieves this object:

```
<metDecl pattern= '((1|0)+\|?/?)*' >
   <symbol value='1'>metrical promimence
   <symbol value='0'>metrical non-prominence
   <symbol value='|'>foot boundary
   <symbol value='/'>metrical line boundary
</metDecl>
```

The same notation might also be specified less formally, as follows:

```
<metDecl>
 <p>Metrically prominent syllables are marked '1' and other
    syllables '0'. Foot divisions are marked by a vertical bar,
    and line divisions with a solidus.
 <p>This notation may be applied to any metrical unit, of any
    size (including, for example, individual feet as well as
    groups of lines).
</metDecl>
```

Note that in this case, because the pattern attribute has not been supplied, no processor can validate met attribute values within the text which use this metrical notation.

For more complex cases, it will often be more convenient to define a notation incrementally. The terminal attribute should be used to indicate for a given symbol whether or not it may be re-defined in terms of other symbols used within the same notation. For example, here is a notation for encoding classical metres, in which symbols are provided for the most common types of foot. These symbols are themselves documented within the same notation, in terms of more primitive long and short syllables:

```
<metDecl pattern='[DTIS3A]+'>
 <symbol value='D' n='dactyl'   terminal=N>-oo
 <symbol value='T' n='trochee'  terminal=N>-o
 <symbol value='I' n='iamb'     terminal=N>o-
 <symbol value='S' n='spondee'  terminal=N>--
 <symbol value='3' n='tribrach' terminal=N>ooo
 <symbol value='A' n='anapaest' terminal=N>oo-
 <symbol value='o'>short syllable
 <symbol value='-'>long syllable
</metDecl>
```

Note here the use of the global n attribute to supply an additional name for the symbols being documented.

For further discussion of this metrical notation and its use in the encoding of verse, see section 9.4 ('Rhyme and Metrical Analysis') on p. 221.

The elements discussed in this section are defined as follows:

```
<!-- 5.3.8:  Metrical Notation Declaration                    -->
<!ELEMENT metDecl       - O  ((%component.seq) | (symbol+))     >
<!ATTLIST metDecl            %a.global;
                             %a.declarable;
          type               NAMES               "MET REAL"
          pattern            CDATA               #IMPLIED       >
<!ELEMENT symbol        - O  (%phrase.seq;)                      >
<!ATTLIST symbol             %a.global;
          value              CDATA               #REQUIRED
          terminal           (Y | N)             Y              >
<!-- This fragment is used in sec. 5.3                          -->
```

5.3.9 The Variant-Encoding Method Element

The <variantEncoding> element is the last of the nine optional subdivisions of the <encodingDesc> element. It is used to document the method used to encode textual variants in the

text, as discussed in section 19.2 ('Linking the Apparatus to the Text') on p. 480.

<variantEncoding> declares the method used to encode text-critical variants. Attributes include:

method indicates which method is used to encode the apparatus of variants. Legal values are:

location-referenced apparatus uses line numbers or other canonical reference scheme referenced in a base text.

double-end-point apparatus indicates the precise locations of the beginning and ending of each lemma relative to a base text.

parallel-segmentation alternate readings of a passage are given in parallel in the text; no notion of a base text is necessary.

location indicates whether the apparatus appears within the running text or external to it. Legal values are:

internal apparatus appears within the running text.

external apparatus appears outside the base text.

Its formal declaration is as follows:

```
<!-- 5.3.9:  Variant-Encoding Declaration                    -->
<!ELEMENT variantEncoding
                          - O  EMPTY                          >
<!ATTLIST variantEncoding    %a.global;
          method             (location-referenced |
                             double-end-point |
                             parallel-segmentation)
                                                 #REQUIRED
          location           (internal | external)
                                                 #REQUIRED    >
<!-- This fragment is used in sec. 5.3                        -->
```

5.4 The Profile Description

The <profileDesc> element is the third major subdivision of the TEI Header. It is an optional element, the purpose of which is to enable information characterizing various descriptive aspects of a text or a corpus to be recorded within a single unified framework.

<profileDesc> provides a detailed description of non-bibliographic aspects of a text, specifically the languages and sublanguages used, the situation in which it was produced, the participants and their setting.

In principle, almost any component of the header might be of importance as a means of characterizing a text. The author of a written text, its title or its date of publication, may all be regarded as characterizing it at least as strongly as any of the parameters discussed in this section. The rule of thumb applied has been to exclude from discussion here most of the information which generally forms part of a standard bibliographic style description, if only because such information has already been included elsewhere in the TEI header.

The core <profileDesc> element has three optional components, represented by the following elements:

<creation> contains information about the creation of a text.

<langUsage> describes the languages, sublanguages, registers, dialects etc. represented within a text.

<textClass> groups information which describes the nature or topic of a text in terms of a standard classification scheme, thesaurus, etc.

These elements are further described in the remainder of this section.

Three other elements may also appear within the <profileDesc> element, when the additional tag set for the TEI header is in use:

<textDesc> provides a description of a text in terms of its *situational parameters*.

<particDesc> describes the identifiable speakers, voices or other participants in a linguistic interaction.

<settingDesc> describes the setting or settings within which a language interaction takes place, either as a prose description or as a series of <setting> elements.

For descriptions of these elements, see section 23.2 ('Contextual Information') on p. 540.

Finally, the following element can appear in the <profileDesc> element, when the additional tag set for transcription of primary sources is selected:

<handList> contains a series of <hand> elements listing the different hands of the source.

For a description of this element, see section 18.2.1 ('Document Hands') on p. 456.

The profile description itself has the following formal definition:

```
<!-- 5.4:  The profile description                          -->
<!ELEMENT profileDesc   - -  (creation?, langUsage*, textDesc*,
                              particDesc*, settingDesc*,
                              handList*, textClass*)            >
<!ATTLIST profileDesc        %a.global;                         >
<!-- ... declarations from section 5.4.1                     -->
<!--     (Creation)                                           -->
<!--     go here ...                                          -->
<!-- ... declarations from section 5.4.2                     -->
<!--     (Language usage)                                     -->
<!--     go here ...                                          -->
<!-- ... declarations from section 5.4.3                     -->
<!--     (Text Classification)                                -->
<!--     go here ...                                          -->
<!-- This fragment is used in sec. 5.1.1                     -->
```

5.4.1 Creation

The <creation> element contains phrases describing the origin of the text, e.g. the date and place of its composition.

<creation> contains information about the creation of a text.

The date and place of composition are often of particular importance for studies of linguistic variation; since such information cannot be inferred with confidence from the bibliographic description of the copy text, the <creation> element may be used to provide a consistent location for this information:

```
<creation>
   <date value='1992-08'>August 1992</date>
   <rs type=city>Taos, New Mexico</rs>
</creation>
```

The formal declaration of <creation> is as follows:

```
<!-- 5.4.1:  Creation                                        -->
<!ELEMENT creation      - o  (%phrase.seq;)                   >
<!ATTLIST creation           %a.global;                       >
<!-- This fragment is used in sec. 5.4                       -->
```

5.4.2 Language Usage

The <langUsage> element is used within the <profileDesc> element to describe the languages, sublanguages, registers, dialects, etc. represented within a text. It contains one or more <language> elements, each of which takes attributes specifying the *writing system* used (see section 4 ('Characters and Character Sets') on p. 71) and the quantity of that language present in the text. Following the <language> elements, prose description may also be added to specify further relevant information.

<langUsage> describes the languages, sublanguages, registers, dialects etc. represented within a text.

<language> characterizes a single language or sublanguage used within a text. Attributes include:

id specifies the identifier for the *writing system declaration* for this language (e.g. *eng*, *fra*, *deu*)

wsd specifies the entity containing the *writing system declaration* used for representing texts in this language.

usage specifies the approximate percentage (by volume) of the text which uses this language.

Each **<language>** element links the document to the formal writing system declaration defining that language and its script; for that reason, its use is recommended. The **wsd** attribute must give the name of an entity containing a writing system declaration; typically, this will be an external file declared in the document type declaration. For example,

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
  <!ENTITY mhd system 'mhd.wsd'>
  <!ENTITY nhd system 'nhd.wsd'>
]>
<tei.2>
<teiHeader> ...
    <profileDesc>
         <langUsage>
              <language id=GMH wsd=mhd>Middle High German
              <language id=DEU wsd=nhd>Modern standard German
         </langUsage>
     ...
</tei.2>
```

When two sublanguages share the same language code and writing system declaration but are distinguished in the **<langUsage>** element, only one of the **<language>** elements should bear the **id** attribute:

```
<langUsage>
   <language id=FR wsd=wsd.fr usage=60>Qu&eacute;becois
   <language id=EN wsd=wsd.en usage=20>Canadian business English
   <language       wsd=wsd.en usage=20>British English
</langUsage>
```

or, less formally,

```
   <langUsage>
   <language id=FR wsd=wsd.fr>
   <language id=EN wsd=wsd.en>
   <p>Approximately 60% of the text is in
     Qu&eacute;becois, the remainder being equally
     divided between Canadian business English and
     British English.
   </langUsage>
```

The **<langUsage>** and **<language>** elements have the following formal definitions:

```
<!-- 5.4.2:  Language usage                                    -->
<!ELEMENT langUsage     - o  (p | language)+                    >
<!ATTLIST langUsage          %a.global;
                             %a.declarable;                     >
<!ELEMENT language      - o  (%phrase.seq)                      >
<!ATTLIST language
          n                  CDATA              #IMPLIED
          lang               IDREF              %INHERITED
          rend               CDATA              #IMPLIED
          id                 ID                 #IMPLIED
          wsd                ENTITY             #IMPLIED
          usage              NUMBER             #IMPLIED       >
<!-- This fragment is used in sec. 5.4                         -->
```

5.4.3 The Text Classification

The second component of the core <profileDesc> element is the <textClass> element. This element is used to classify a text according to one or more of the following methods:

- by reference to a recognized international classification such as the Dewey Decimal Classification, the Universal Decimal Classification, the Colon Classification, the Library of Congress Classification, or any other system widely used in library and documentation work
- by providing a set of keywords, as provided for example by British Library or Library of Congress Cataloguing in Publication data
- by referencing any other taxonomy of text categories recognized in the field concerned, or peculiar to the material in hand; this may include one based on recurring sets of values for the situational parameters defined in section 23.2.1 ('The Text Description') on p. 541, or the demographic elements described in section 23.2.2 ('The Participants Description') on p. 545

The last of these may be particularly important for dealing with existing corpora or collections, both as a means of avoiding the expense or inconvenience of reclassification and as a means of documenting the organizing principles of such materials.

The following tags are provided for this purpose:

<keywords> contains a list of keywords or phrases identifying the topic or nature of a text. Attributes include:

scheme identifies the controlled vocabulary within which the set of keywords concerned is defined.

<classCode> contains the classification code used for this text in some standard classification system. Attributes include:

scheme identifies the classification system or taxonomy in use.

<catRef> specifies one or more defined categories within some taxonomy or text typology. Attributes include:

target identifies the categories concerned

The <keywords> element simply categorizes an individual text by supplying a list of keywords which may describe its topic or subject matter, its form, date, etc. In some schemes, the order of items in the list is significant, for example, from major topic to minor; in others, the list has an organized substructure of its own. No recommendations are made here as to which method is to be preferred. Wherever possible, such keywords should be taken from a recognized source, such as the British Library/Library of Congress Cataloguing in Publication data in the case of printed books, or a published thesaurus appropriate to the field.

The scheme attribute should be used to indicate the source of the keywords used. This is done by supplying the value used for the id attribute of a <taxonomy> element within which further details of the source concerned may be found. The <taxonomy> element occurs in the <classDecl> part of the encoding declarations within the TEI Header and is described in section 5.3.6 ('The Classification Declaration') on p. 104. For example:

```
<keywords scheme=LCSH>
  <list><item>Data base management</item>
        <item>SQL (Computer program language)</item>
  </list>
</keywords>

<keywords scheme=LCSH>
  <list>
  <item>English literature -- History and criticism --
        Data processing.</item>
  <item>English literature -- History and criticism --
        Theory, etc.</item>
  <item>English language -- Style -- Data processing.</item>
  <item>Style, Literary -- Data processing.</item>
  </list>
</keywords>
```

The <classCode> element also categorizes an individual text, by supplying a numerical or other code used in a recognized classification scheme, such as the Dewey Decimal Classification. The scheme attribute is used to indicate the source of the classification scheme, in the same way as for the <keywords> element, as in the following example:

```
<classCode scheme=DDC19>005.756</>
<classCode scheme=LC>QA76.9</>
<classCode scheme=DDC19>820.285</>
<classCode scheme=LC>PR21</>
```

The <catRef> element categorizes an individual text by pointing to one or more <category> elements. The <category> element (which is fully described in section 5.3.6 ('The Classification Declaration') on p. 104) holds information about a particular classification or category within a given taxonomy. Each such category must have a unique identifier, which may be supplied as the value of the target attribute for <catRef> elements which are regarded as falling within the category indicated.

A text may, of course, fall into more than one category, in which case more than one identifier will be supplied as the value for the target attribute on the <catRef> element, as in the following example:

```
<catRef target="B1 B2 B5">
```

Where more than one descriptive taxonomy is used to characterize the texts in a corpus or collection, the scheme attribute should be supplied to specify the taxonomy to which the categories identified by the target attribute belong. For example,

```
<catRef scheme='Brown' target='B12 B15'>
<catRef scheme='SUC' target='A45'>
```

Here the same text has been classified as of categories "B12" and "B15" within the Brown classification scheme, and as of category "A45" within the SUC classification scheme.

The distinction between the <catRef> and <classCode> elements is that the values used as identifying codes *must* be defined somewhere within the header for the former, but not the latter.

The elements described in this section have the following formal definitions:

```
<!-- 5.4.3:  Text Classification                                  -->
<!ELEMENT textClass     - -  ((classCode | catRef | keywords)*
                              )                                    >
<!ATTLIST textClass          %a.global;
                             %a.declarable;                        >
<!ELEMENT keywords      - o  (term+ | list)                        >
<!ATTLIST keywords           %a.global;
          scheme             IDREF               #IMPLIED         >
<!ELEMENT classCode     - -  (%phrase.seq)                         >
<!ATTLIST classCode          %a.global;
          scheme             IDREF               #IMPLIED         >
<!ELEMENT catRef        - o  EMPTY                                 >
<!ATTLIST catRef             %a.global;
          target             IDREFS              #REQUIRED
          scheme             IDREF               #IMPLIED         >
<!-- This fragment is used in sec. 5.4                           -->
```

5.5 The Revision Description

The final subelement of the TEI header, the <revisionDesc> element, provides a detailed change log in which each change made to a text may be recorded. Its use is optional but highly recommended. It provides essential information for the administration of large numbers of files which are being updated, corrected, or otherwise modified as well as extremely useful documentation for files being passed from researcher to researcher or system to system. Without change logs, it is easy to confuse different versions of a file, or to remain unaware of small but

important changes made in the file by some earlier link in the chain of distribution. No change should be made in any TEI-conformant file without corresponding entries being made in the change log.

<revisionDesc> summarizes the revision history for a file.
<change> summarizes a particular change or correction made to a particular version of an electronic text which is shared between several researchers.

The log consists of a list of entries, one for each change. This may be encoded using either the regular <list> element, as described in section 6.7 ('Lists') on p. 149 or as a series of special purpose <change> elements, each of which has the following constituents:

<date> contains a date in any format.
<respStmt> supplies a statement of responsibility for someone responsible for the intellectual content of a text, edition, recording, or series, where the specialized elements for authors, editors, etc. do not suffice or do not apply.
<item> contains one component of a list.

The <date> element indicates the date of the change. The <respStmt> element indicates who made the change, and in what role. The <item> element indicates what change was made; it can range from a simple phrase to a series of paragraphs. If a number is to be associated with one or more changes (for example, a revision number), use the global n attribute on the <change> element to supply it.

It is recommended to give changes in reverse chronological order, most recent first.
For example:

```
<revisionDesc>
<change><date>6/3/91:</date>
        <respStmt><name>EMB</name><resp>ed.</></respStmt>
        <item> < > changes completed.</item>
<change><date>5/25/91:</date>
        <respStmt><name>EMB</name><resp>ed.</></respStmt>
        <item>File format updated.</item>
<change><date>9/3/90:</date>
        <respStmt><name>EMB</name><resp>ed.</></respStmt>
        <item>Changes to make a prettier printed version.</item>
<change><date>5/25/90:</date>
        <respStmt><name>EMB</name><resp>ed.</></respStmt>
        <item>Stuart's corrections entered</item>
<change><date>2/90: </date>
        <respStmt><name>N.N.</name><resp>admin.</></respStmt>
        <item>Sent to Stuart Curran for proofreading.</item>
<change><date>1/22/90:</date>
        <respStmt><name>N.N.</name><resp>data entry</></respStmt>
        <item>Corrections made to file;</item>
<change><date>10/89:</date>
        <respStmt><name>John G. Fitzgerald, Julia M. Deisler
            and Deborah Hirsch.</name><resp>staff</></respStmt>
        <item>Proofread.</item>
<change><date>8/89:</date>
        <respStmt><name>Amy E. Frisch</><resp>data entry</></respStmt>
        <item>Input begun</item>
</revisionDesc>
```

The formal definition of the <revisionDesc> element is thus as follows:

```
<!-- 5.5:  The Revision Description                          -->
<!ELEMENT revisionDesc  - -  (list | change+)                 >
<!ATTLIST revisionDesc       %a.global;                       >
<!ELEMENT change        - O  (date, respStmt+, item)          >
<!ATTLIST change             %a.global;                       >
<!-- respStmt, item, and date are declared in teicore2.      -->
```

```
<!-- This fragment is used in sec. 5.1.1                    -->
```

5.6 Minimal and Recommended Headers

The TEI header allows for the provision of a very large amount of information concerning the text itself, its source, encodings and revisions of it, as well as a wealth of descriptive information such as the languages it uses and the situation within which it was produced, the setting and identity of participants within it. This diversity and richness reflects the diversity of uses to which it is envisaged that electronic texts conforming to these Guidelines will be put. It is emphatically *not* intended that all of the elements described above should be present in every TEI Header.

The amount of encoding in a header will depend both on the nature and the intended use of the text. At one extreme, an encoder may expect that the header will be needed only to provide a bibliographic identification of the text adequate to local needs. At the other, wishing to ensure that their texts can be used for the widest range of applications, encoders will want to document as explicitly as possible both bibliographic and descriptive information, in such a way that no prior or ancillary knowledge about the text is needed in order to process it. The header in such a case will be very full, approximating to the kind of documentation often supplied in the form of a manual. Most texts will lie somewhere between these extremes; textual corpora in particular will tend more to the latter extreme.

For each element discussed above, an indication is given in the general alphabetical index (section VII ('Alphabetical Reference List of Tags and Attributes') on p. 647) as to whether its encoding is regarded in general as required, recommended or optional. Clearly, all elements relating to descriptive matters such as text classification or description must be optional in the general case, though for certain kinds of analysis their presence will be mandatory. This section therefore confines itself to demonstrating the minimal and recommended levels of encoding of the bibliographic information held by the TEI header.

Supplying only the minimal level of encoding required, the TEI header of a single printed text might look like the following example:

```
<teiHeader>
  <fileDesc>
    <titleStmt>
        <title>Thomas Paine: Common sense, a
            machine-readable transcript</title>
        <respStmt><resp>compiled by</resp>
             <name>Jon K Adams</name>
        </respStmt>
     </titleStmt>
     <publicationStmt>
        <distributor>Oxford Text Archive</distributor>
     </publicationStmt>
     <sourceDesc>
        <bibl>The complete writings of Thomas Paine,
           collected and edited by Phillip S. Foner
           (New York, Citadel Press, 1945)
        </bibl>
      </sourceDesc>
   </fileDesc>
<teiHeader>
```

The only mandatory component of the TEI Header is the <fileDesc> element. Within this, <titleStmt>, <publicationStmt> and <sourceDesc> are all required constituents. Within the title statement, a title is required, and an author should be specified, even if it is unknown, as should some additional statement of responsibility, here given by the <respStmt> element. Within the <publicationStmt>, a publisher, distributor or other agency responsible for the file must be specified. Finally, the source description should contain at the least a loosely structured

bibliographic citation identifying the source of the electronic text if (as is usually the case) there is one.

We now present the same example header, expanded to include additionally recommended information, adequate to most bibliographic purposes, in particular to allow for the creation of an AACR2-conformant bibliographic record. We have also added information about the encoding principles used in this (imaginary) encoding, about the text itself (in the form of Library of Congress subject headings), and about the revision of the file.

```
<teiHeader>
<fileDesc>
    <titleStmt>
         <title>Common sense, a machine-readable transcript</title>
         <author>Paine, Thomas (1737-1809)</author>
         <respStmt><resp>compiled by</resp>
               <name>Jon K Adams</name>
         </respStmt>
    </titleStmt>
    <editionStmt>
         <edition><date>1986</date></edition>
    </editionStmt>
    <publicationStmt>
         <distributor>Oxford Text Archive.</distributor>
         <address>Oxford University Computing Services,
                  13 Banbury Road, Oxford OX2 6RB, UK
    </publicationStmt>
    <notesStmt>
         <note>Brief notes on the text are in a
               supplementary file.</note>
    </notesStmt>
    <sourceDesc>
         <biblStruct>
              <editor>Foner, Philip S.</editor>
              <title>The collected writings of Thomas Paine</title>
              <imprint>
                   <pubPlace>New York</pubPlace>
                   <publisher>Citadel Press</publisher>
                   <date>1945</date>
              </imprint>
         </biblStruct>
    </sourceDesc>
</fileDesc>
<encodingDesc>
    <samplingDecl><p>Editorial notes in the Foner edition have not
         been reproduced.
         <p>Blank lines and multiple blank spaces, including paragraph
         indents, have not been preserved.
    <editorialDecl>
         <correction status=high method=silent><p>The following errors
              in the Foner edition have been corrected:
              <list>
              <item>p.  13 l.  7 cotemporaries   contemporaries
              <item>p.  28 l. 26 [comma]         [period]
              <item>p.  84 l.  4 kin             kind
              <item>p.  95 l.  1 stuggle         struggle
              <item>p. 101 l.  4 certainy        certainty
              <item>p. 167 l.  6 than            that
              <item>p. 209 l. 24 publshed        published
              </list>
         <normalization><p>No normalization beyond that performed
              by Foner, if any.
         <quotation marks=all form=std><p>All double quotation marks
```

```
                rendered with ", all single quotation marks with
                apostrophe.
           <hyphenation eol=none><p>Hyphenated words that appear at the
                end of the line in the Foner edition have been reformed.
           <stdVals><p>Standard date values are given in ISO form:
                yyyy-mm-dd.
           <interpretation><p>Compound proper names are marked.
                <p>Dates are marked.
                <p>Italics are recorded without interpretation.
      </editorialDecl>
      <classDecl>
           <taxonomy id='LCSH'>
                <bibl>Library of Congress Subject Headings</></>
           <taxonomy id='LC'>
                <bibl>Library of Congress Classification</></>
      </classDecl>
  </encodingDesc>
  <profileDesc>
      <creation><date>1774</date></creation>
      <langUsage>
           <language id=EN wsd='english' usage=100>English.</language>
      </langUsage>
      <textClass>
           <keywords scheme='LCSH'>
                <list>
                     <item>Political science</item>
                     <item>United States -- Politics and government --
                          Revolution, 1775-1783</item>
                </list>
           </keywords>
           <classCode scheme='LC'>JC 177</>
      </textClass>
  </profileDesc>
  <revisionDesc>
  <change><date>1996-01-22 <name>CMSMcQ<what>finished proofreading</change>
  <change><date>1995-10-30 <name>L.B. <what>finished proofreading</change>
  <change><date>1995-07-20 <name>R.G. <what>finished proofreading</change>
  <change><date>1995-07-04 <name>R.G. <what>finished data entry</change>
  <change><date>1995-01-15 <name>R.G. <what>began data entry</change>
  </revisionDesc>
  </teiHeader>
```

Many other examples of recommended usage for the elements discussed in this chapter are provided here, in the reference index and in the associated tutorials.

5.7 Note for Library Cataloguers

A strong motivation in preparing the material in this chapter was to provide in the TEI file header a viable chief source of information for cataloguing the machine-readable data file. The file header is not a library catalogue record, and so will not make all of the distinctions essential in standard library work. It also includes much information generally excluded from standard bibliographic descriptions. It is the intention of the developers, however, to ensure that the information required for a catalogue record be retrievable from the TEI file header, and moreover that the mapping from the one to the other be as simple and straightforward as possible. Where the correspondence is not obvious, it may prove useful to consult one of the works which were influential in developing the content of the TEI file header. These include:

ISBD(G) The International Standard Book Description (General) is an international standard setting out what information should be recorded in a description of a bibliographical item.

There are also separate ISBDs covering different types of material, e.g. ISBD(M) for monographs, ISBD(CF) for computer files. These separate ISBDs follow the same general scheme as the main ISBD(G), but provide appropriate interpretations for the specific materials under consideration.

AACR2 The Anglo-American Cataloguing Rules (second edition) were published in 1978, with a revision appearing in 1988. The AACR2 provides guidelines for the construction of catalogues in general libraries. AACR2 is explicitly based on the general framework of the ISBD(G), and the subsidiary ISBDs. It gives a description of how to catalogue items according to the ISBDs, and how to construct indexes and catalogue cross references.

ANSI Z.39.29 ANSI Z.39.29 is an American national standard governing bibliographic references for use in bibliographies, end-of-work lists, references in abstracting and indexing publications, and outputs from computerized bibliographic data bases. This standard has however now been withdrawn, pending substantial revision. The international standard which covers the same area is ISO 690: 1987. Other relevant standards include BS 1629: 1989, BS 5605: 1978, and BS 6371:1983.

Chapter 6

Elements Available in All TEI Documents

This chapter describes elements which may appear in any kind of text and the tags used to mark them in all TEI documents. Most of these elements are freely floating phrases, which can appear at any point within the textual structure, although they must generally be contained by a higher-level element of some kind (such as a paragraph). A few of the elements described in this chapter (for example, bibliographic citations and lists) have a comparatively well-defined internal structure, but most of them have no consistent inner structure of their own. In the general case, they contain only a few words, and are often identifiable in a conventionally printed text by the use of typographic conventions such as shifts of font, use of quotation or other punctuation marks, or other changes in layout.

To use the terminology introduced in section 3.7.3 ('The TEICLAS2.ENT File') on p. 55, most of the elements described in this chapter are members of the class *phrase*, and a small number are members of the classes *chunk* or *inter*.

This chapter begins by describing the <p> tag used to mark paragraphs, which serve as the fundamental formal unit for running text in many base tag sets, and are available in all. This is followed, in section 6.2 ('Treatment of Punctuation') on p. 121, by a discussion of some specific problems associated with the interpretation of conventional punctuation, and the methods proposed by the current Guidelines for resolving ambiguities therein.

The next section (section 6.3 ('Highlighting and Quotation') on p. 123) describes a number of phrase-level elements commonly marked by typographic features (and thus well-represented in conventional markup languages). These include features commonly marked by font shifts (section 6.3.2 ('Emphasis, Foreign Words, and Unusual Language') on p. 124) and features commonly marked by quotation marks (section 6.3.3 ('Quotation') on p. 127) as well as such features as terms, cited words, and glosses (section 6.3.4 ('Terms, Glosses, and Cited Words') on p. 130).

The next section (section 6.4 ('Names, Numbers, Dates, Abbreviations, and Addresses') on p. 132) describes several phrase-level and inter-level elements which, although often of interest for analysis or processing, are rarely explicitly identified in conventional printing. These include names (section 6.4.1 ('Referring Strings') on p. 132), numbers and measures (section 6.4.3 ('Numbers and Measures') on p. 135), dates and times (section 6.4.4 ('Dates and times') on p. 137), abbreviations (section 6.4.5 ('Abbreviations and Their Expansions') on p. 139), and addresses (section 6.4.2 ('Addresses') on p. 134).

Section 6.5 ('Simple Editorial Changes') on p. 140 introduces some phrase-level elements which may be used to record simple editorial emendation or correction of the encoded text. The tags described here constitute a simple subset of the full mechanisms for encoding such information (described in full in chapter 18 ('Transcription of Primary Sources') on p. 443), which should be adequate to most commonly encountered situations.

In the same way, the following section (section 6.6 ('Simple Links and Cross References') on p. 147) presents only a subset of the facilities available for the encoding of cross-references or text-linkage. The full story may be found in chapter 14 ('Linking, Segmentation, and Alignment') on

p. 331; the tags presented here are intended to be usable for a wide variety of simple applications.

Sections 6.7 ('Lists') on p. 149, and 6.8 ('Notes, Annotation, and Indexing') on p. 152, describe two kinds of quasi-structural elements, lists and notes, which may appear either within chunk-level elements such as paragraphs, or between them. Several kinds of lists are catered for, of an arbitrary complexity. The section on notes discusses both notes found in the source and simple mechanisms for adding annotations of an interpretive nature during the encoding; again, only a subset of the facilities described in full elsewhere (specifically, in chapter 15 ('Simple Analytic Mechanisms') on p. 381) is discussed.

Next, section 6.9 ('Reference Systems') on p. 155, describes methods of encoding within a text the conventional system or systems used when making references to the text. Some reference systems have attained canonical authority and must be recorded to make the text useable in normal work; in other cases, a convenient reference system must be created by the creator or analyst of an electronic text.

Like lists and notes, the bibliographic citations discussed in section 6.10 ('Bibliographic Citations and References') on p. 162, may be regarded as structural elements in their own right. A range of possibilities is presented for the encoding of bibliographic citations or references, which may be treated as simple phrases within a running text, or as highly-structured components suitable for inclusion in a bibliographic database.

Additional elements for the encoding of passages of verse or drama (whether prose or verse) are discussed in section 6.11 ('Passages of Verse or Drama') on p. 176.

The chapter concludes with a technical overview of the structure and organization of the tag set described here. This should be read in conjunction with chapter 3 ('Structure of the TEI Document Type Definition') on p. 35, describing the structure of the TEI document type definition.

6.1 Paragraphs

The paragraph is the fundamental organizational unit for all prose texts, being the smallest regular unit into which prose can be divided. Prose can appear in all TEI texts, not simply in those using the prose base (section 8 ('Base Tag Set for Prose') on p. 211); the paragraph is therefore described here, as an element which can appear in any kind of text.

Paragraphs can contain any of the other elements described within this chapter, as well as some other elements which are specific to individual text types. We distinguish *phrase-level* elements, which must be entirely contained within a paragraph and cannot appear except within one, from *chunks*, which can appear between, but not within, paragraphs, and from *inter-level* elements, which can appear either within a single paragraph or between paragraphs. The class of phrases includes emphasized or quoted phrases, names, dates, etc. The class of inter-level elements includes bibliographic citations, notes, lists, etc. The class of chunks includes the paragraph itself.

Because paragraphs may appear in different base or additional tag sets, their possible contents may differ in different kinds of documents. In particular, additional elements not listed in this chapter may appear in paragraphs in certain kinds of text. However, the elements described in this chapter are always by default available in all kinds of text.

The paragraph is marked using the `<p>` element:

`<p>` marks paragraphs in prose.

If a consistent internal subdivision of paragraphs is desired, the `<s>` or `<seg>` ("segment") elements may be used, as discussed in chapters 14 ('Linking, Segmentation, and Alignment') on p. 331 and 15 ('Simple Analytic Mechanisms') on p. 381 respectively. More usually, however, paragraphs have no firm internal structure, but contain prose encoded as a mix of characters, entity references, phrases marked as described in the rest of this chapter, and embedded elements like lists, figures, or tables.

Since paragraphs are usually explicitly marked in Western texts, typically by indentation, the application of the `<p>` tag usually presents few problems.

In some cases, the body of a text may comprise but a single paragraph:

```
<body>
<p>I fully appreciate Gen. Pope's splendid achievements with their
invaluable results; but you must know that Major Generalships in the
Regular Army, are not as plenty as blackberries.
</body>
```

This news story shows typically short journalistic paragraphs:

```
<head>SARAJEVO, Bosnia and Herzegovina, April 19</head>
<p>Serbs seized more territory in this struggling new
country today as the United States Air Force ended a
two-day airlift of humanitarian aid into the capital,
Sarajevo.
<p>International relief workers called on European
Community nations to step up their humanitarian aid to
the former Yugoslav republic, in conjunction with new
American aid flights if necessary.
<p>A special envoy from the European Community, Colin
Doyle, harshly condemned the decision by Serbs to shell
Sarajevo on Saturday night during a visit to the Bosnian
capital by a senior American official, Deputy Assistant
Secretary of State Ralph R. Johnson.
<p>...
```

The following extract from a Russian fairy tale demonstrates how other phrase level elements (in this case <q> elements representing direct speech; see section 6.3.3 ('Quotation') on p. 127) may be nested within, but not across, paragraphs:

```
<p>A fly built a castle, a tall and mighty castle.
There came to the castle the Crawling Louse.  <q>Who,
who's in the castle?  Who, who's in your house?</q>
said the Crawling Louse.  <q>I, I, the Languishing Fly.
And who art thou?</q> <q>I'm the Crawling Louse.</q>

<p>Then came to the castle the Leaping Flea.  <q>Who,
who's in the castle?</q> said the Leaping Flea.  <q>I,
I, the Languishing Fly, and I, the Crawling Louse.  And
who art thou?</q> <q>I'm the Leaping Flea.</q>

<p>Then came to the castle the Mischievous Mosquito.
<q>Who, who's in the castle?</q> said the Mischievous
Mosquito.  <q>I, I, the Languishing Fly, and I, the
Crawling Louse, and I, the Leaping Flea.  And who art
thou?</q> <q>I'm the Mischievous Mosquito.</q>
```

The <p> element is formally declared as follows:

```
<!-- 6.1:  Paragraph                                          -->
<!ELEMENT p             - O  (%paraContent;)                  >
<!ATTLIST p                  %a.global;                       >
<!-- This fragment is used in sec. 6.12                       -->
```

6.2 Treatment of Punctuation

Punctuation marks cause problems for text markup because they may not be available in the character set used and because they are often ambiguous. In the former case entity names should be used to render the punctuation mark (see 4 ('Characters and Character Sets') on p. 71). In the latter case, ambiguous punctuation may be treated as described below.

Full stop (period) may mark (orthographic) sentence boundaries, abbreviations, decimal points, or serve as a visual aid in printing numbers. These usages can be distinguished by tagging S-units,

abbreviations, and numbers, as described in sections 14.3 ('Segments and Anchors') on p. 355, 6.4.5 ('Abbreviations and Their Expansions') on p. 139, and 6.4.3 ('Numbers and Measures') on p. 135. There are independent reasons for tagging these, whether or not they are marked by full stops. Alternatively, the following TEI-specific entity names may be used to distinguish stops (and other characters) used for these purposes:

stop.abbr a stop used to end an abbreviation
stop.sent a stop used to end a sentence
stop.abse a stop used both to end an abbreviation and to end a sentence
stop.dec a stop used as a decimal point
comma.dec a comma used as a decimal point
middot.dec a midline dot used as a decimal point
stop.space a stop used as a numeric space character
comma.space a comma used as a numeric space character

These entities are defined in the file *teipunc2*, which is documented in chapter 37 ('Obtaining TEI WSDs') on p. 985.

Question mark and *exclamation mark* typically mark the end of orthographic sentences, but may also be used as a mid-sentence comment by the author ('!' to express surprise or some other strong feeling, '?' to query a word or expression or mark a sentence as dubious in linguistic discussion). These uses may be distinguished by marking S-units, in which case the mid-sentence uses of these punctuation marks may be left unmarked.

Hyphens

at line-end may or may not indicate permanent ("hard") hyphens in the word. Where the lineation of the machine-readable text differs from the original, the editor may eliminate soft (line-end) hyphens or replace them by a reference to the entity *shy* ("soft hyphen"), which is defined in the standard public entity set *ISOnum* defined in ISO 8879 (which should be invoked in the DTD subset if the entity *shy* is to be used). The solution chosen should be reported in the <hyphenation> tag of the encoding declarations in the TEI header. See chapter 5 ('The TEI Header') on p. 77 for discussion of the TEI header and encoding declarations.

Creators of machine-readable texts are recommended to avoid soft hyphens, as one cannot tell whether the hyphens are soft or hard in the case of compounds or prefixed words which might or might not be hyphenated in mid-sentence.

Dashes are best distinguished in form by using the entity names provided in the public entity set *ISOpub*, defined in ISO 8879: *mdash*, *ndash*, and *dash* (the "true" hyphen). Dashes are used for a variety of purposes: insertion, interruption, new speaker (in dialogue), list item. In the latter two cases it is preferable to mark the underlying feature using the elements <q> or <item>, on which see section 6.3.3 ('Quotation') on p. 127, and section 6.7 ('Lists') on p. 149, respectively.

Quotation marks should generally be replaced by the tags <q> or <quote>, especially as quotations are not always marked by quotation marks (notably long quotations) or may be marked in a variety of ways; see the discussion of quotation and related features in section 6.3.3 ('Quotation') on p. 127.

Apostrophes must be distinguished from single quote marks. This is best done by tagging quotations or other uses of quotation marks (see above). However, apostrophes have a variety of uses. In English they mark contractions, genitive forms, and (occasionally) plural forms. Full disambiguation of these uses belongs to the level of linguistic analysis and interpretation.

Parentheses and other marks of suspension such as dashes or ellipses are often used to signal information about the syntactic structure of a text fragment. Full disambiguation of their uses also belongs to the level of linguistic analysis and interpretation, and is therefore discussed in chapter 15 ('Simple Analytic Mechanisms') on p. 381.

Where punctuation marks are disambiguated by tagging the underlying feature they signal, it may be debated whether they should be excluded or left as part of the text. In the case of quotation marks, it may sometimes be more convenient to distinguish opening from closing marks simply by using the appropriate entity reference, rather than using the <q> element, with or without a rend attribute. The solution chosen will vary depending upon the feature and depending upon the purpose of the project.

6.3 Highlighting and Quotation

This section deals with a variety of textual features, all of which have in common that they are frequently realized in conventional printing practice by the use of such features as underlining, italic fonts, or quotation marks, collectively referred to here as *highlighting*. After an initial discussion of this phenomenon and alternate approaches to encoding it, this section describes ways of encoding the following textual features, all of which are conventionally rendered using some kind of highlighting:

- emphasis, foreign words and other linguistically distinct uses of highlighting
- representation of speech and thought, quotation, etc.
- technical terms, glosses, etc.

6.3.1 What Is Highlighting?

By 'highlighting' we mean the use of any combination of typographic features (font, size, hue, etc.) in a printed or written text in order to distinguish some passage of a text from its surroundings.[1] The purpose of highlighting is generally to draw the reader's attention to some feature or characteristic of the passage highlighted; this section describes the elements recommended by these Guidelines for the encoding of such textual features.

In conventionally printed modern texts, highlighting is often employed to identify words or phrases which are regarded as being one or more of the following:

- distinct in some way — as foreign, dialectal, archaic, technical, etc.
- emphatic, and which would for example be stressed when spoken
- not part of the body of the text, for example cross references, titles, headings, labels, etc.
- identified with a distinct narrative stream, for example an internal monologue or commentary.
- attributed by the narrator to some other agency, either within the text or outside it: for example, direct speech or quotation.
- set apart from the text in some other way: for example, proverbial phrases, words mentioned but not used, names of persons and places in older texts, editorial corrections or additions, etc.

The textual functions signalled by highlighting may not be rendered consistently in different parts of a text or in different texts. (For example, a foreign word may appear in italics if the surrounding text is in roman, but in roman if the surrounding text is in italics.) For this reason, these Guidelines distinguish between the encoding of rendering itself and the encoding of the underlying feature expressed by it.

Highlighting as such may be encoded by using the global `rend` attribute which can be specified for any element in the TEI scheme. This allows the encoder both to specify the function of a highlighted phrase or word, by selecting the appropriate element described here or elsewhere in the Guidelines, and to further describe the way in which it is highlighted, by means of the `rend` attribute. If the encoder wishes to offer no interpretation of the feature underlying the use of highlighting in the source text, then the `<hi>` element may be used, which indicates only that the text so tagged was highlighted in some way.

The possible values carried by the `rend` attribute are not formally defined in this version of the Guidelines. Since the `rend` attribute may be used to document any peculiarity of the way a given segment of text was rendered in the original source text, it may need to express a very large range of typographic features, by no means restricted to type face, type size, etc.

Where it is both appropriate and feasible, these Guidelines recommend that the textual feature marked by the highlighting should be encoded, rather than just the simple fact of the highlighting. This is for the following reasons:

- the same kind of highlighting may be used for different purposes in different contexts
- the same textual function may be highlighted in different ways in different contexts

[1] Although the way in which a spoken text is performed, (for example, the voice quality, loudness, etc.) might be regarded as analogous to "highlighting" in this sense, these Guidelines recommend distinct elements for the encoding of such "highlighting" in spoken texts. See further section 11.2.6 ('Shifts') on p. 258.

- for analytic purposes, it is in general more useful to know the intended function of a highlighted phrase than simply that it is distinct.

In many, if not most, cases the underlying function of a highlighted phrase will be obvious and non-controversial, since the distinctions indicated by a change of highlighting correspond with distinctions discussed elsewhere in these Guidelines. It should be recognized, however, that cases do exist in which is it not economically feasible to mark the underlying function of highlighting (e.g. in the preparation of large text corpora), as well as cases in which it is not intellectually appropriate (as in the transcription of some older materials, or in the preparation of material for the study of typographic practice). In such cases, the <hi> element should be used, as further discussed below.

Elements which are sometimes realized by typographic distinction but which are not discussed in this section include <title> (discussed in section 6.10 ('Bibliographic Citations and References') on p. 162) and <name> (discussed in section 6.4.1 ('Referring Strings') on p. 132).

6.3.2 Emphasis, Foreign Words, and Unusual Language

This subsection discusses the following elements:

<foreign> identifies a word or phrase as belonging to some language other than that of the surrounding text. Attributes include:

lang identifies the language of the word or phrase marked.

<emph> marks words or phrases which are stressed or emphasized for linguistic or rhetorical effect.

<hi> marks a word or phrase as graphically distinct from the surrounding text, for reasons concerning which no claim is made. Attributes include:

rend describes the rendition or presentation of the word or phrase highlighted.

<distinct> identifies any word or phrase which is regarded as linguistically distinct, for example as archaic, technical, dialectal, non-preferred, etc., or as forming part of a sublanguage. Attributes include:

type specifies the sublanguage or register to which the word or phrase is being assigned
time specifies how the phrase is distinct diachronically
space specifies how the phrase is distinct diatopically
social specifies how the phrase is distinct diastatically

6.3.2.1 Foreign Words or Expressions

Words or phrases which are not in the main language of the text should be tagged as such, at least where the fact is indicated in the text. Where the word or phrase concerned is already distinguished from the rest of the text by virtue of its function (for example, because it is a name, a technical term, a quotation, a mentioned word, etc.) then the global lang attribute should be used to specify additionally that its language distinguishes it from the surrounding text. Any element in the TEI scheme may take a lang attribute, which specifies both the writing system and the language used by its content (see section 4.2 ('Shifting Among Character Sets') on p. 74 for discussion of this attribute). Where there is no other applicable element, the tag <foreign> may be used to provide a peg onto which the lang may be attached.

```
<q>Aren't you confusing <foreign lang=la>post hoc</foreign>
with <foreign lang=la>propter hoc</foreign>?</q> said the
Bee Master.  <q>Wax-moth only succeed when weak bees let
them in.</q>
```

The <foreign> tag should not be used to encode foreign words which are mentioned or glossed within the text: for these use the appropriate element from section 6.3.4 ('Terms, Glosses, and Cited Words') on p. 130 below. Compare the following example sentences:

```
John eats a <foreign lang=fr>croissant</foreign> every morning.
```

```
<mentioned lang=fr>Croissant</mentioned> is difficult to
pronounce with your mouth full.


A <term lang=fr>croissant</term> is a crescent-shaped piece
of light, buttery, pastry that is usually eaten for
breakfast, especially in France.
```

The <foreign> element is formally defined as follows:

```
<!-- 6.3.2.1:  Highlighted phrases                          -->
<!ELEMENT foreign       - -  (%paraContent;)                 >
<!ATTLIST foreign
          id                ID                  #IMPLIED
          n                 CDATA               #IMPLIED
          rend              CDATA               #IMPLIED
          lang              IDREF               #IMPLIED     >
<!-- (continued in sec. 6.3.2.2, 6.3.2.3, 6.3.3, 6.3.4)     -->
<!-- This fragment is used in sec. 6.12                     -->
```

6.3.2.2 Emphatic Words and Phrases

The <emph> element is provided to mark words or phrases which are *linguistically* emphatic or stressed. Text which is only typographically "emphasized" falls into the class of highlighted text, and may be tagged with the <hi> element. In printed works, emphasis is generally indicated by devices such as the use of an italic font, a large typeface or extra wide letter spacing; in manuscripts and typescripts, it is usually indicated by the use of underlining.

As the following examples demonstrate, an encoder may choose whether or not to make explicit the particular type of rendition associated with the emphasis, by use of the rend attribute. If a source text consistently renders a particular feature (e.g. emphasis or words in foreign languages) in a particular way, the rendering associated with that feature may be described in the TEI header and the rend attribute used only to describe examples which deviate from the norm.

```
<q>Sex, sir, is <emph>purely</emph> a question of appetite!</q>
   Tarr exclaimed.

<q>What it all comes to is this,</q> he said.
<q><emph rend=italic>What does Christopher
Robin do in the morning nowadays?</emph></q>

<l>Here Thou, great <name rend=italics>Anna</name>!
      whom three Realms obey,
<l>Doth sometimes Counsel take &mdash;
      and sometimes <emph rend=italic>Tea</emph>.
```

The <hi> element is used to mark words or phrases which are highlighted in some way, but for which identification of the intended distinction is difficult, controversial or impossible. It enables an encoder simply to record the fact of highlighting, possibly describing it by the use of a rend attribute, as discussed above, without however taking a position as to the function of the highlighting. This may also be useful if the text is to be processed in two stages: representing simply typographic distinctions during a first pass, and then replacing the <hi> tags with more specific tags in a second pass.

Some simple examples:

```
<hi rend=gothic>And this Indenture further witnesseth</hi>
that the said <hi rend=italic>Walter Shandy</hi>, merchant,
in consideration of the said intended marriage ...
```

In this example, the first highlighted phrase uses black letter or gothic print to mimic the appearance of a legal document, and italic to mark 'Walter Shandy' as a name. In a second pass, the elements <head> or <label> might be appropriate for the first use, and the element <name> for the second.

```
The heaviest rain, and snow, and hail, and sleet, could
boast of the advantage over him in only one respect.  They
often <hi rend=quoted>came down</hi> handsomely, and
Scrooge never did.
```

In this example, the phrase 'came down' uses inverted commas to indicate a play on words.[2] In a second pass, the element <soCalled> might be preferred.

The <emph> and <hi> elements are formally defined as follows:

```
<!-- 6.3.2.2:  Highlighted phrases (cont'd)                    -->
<!-- (continuation of sec. 6.3.2.1)                            -->
<!ELEMENT emph          - -  (%paraContent;)                     >
<!ATTLIST emph               %a.global;                          >
<!ELEMENT hi            - -  (%paraContent;)                     >
<!ATTLIST hi
          id                 ID                  #IMPLIED
          n                  CDATA               #IMPLIED
          lang               IDREF               %INHERITED
          rend               CDATA               #IMPLIED       >
```

6.3.2.3 Other Linguistically Distinct Material

For some kinds of analysis, it may be desirable to encode the linguistic distinctiveness of words and phrases with more delicacy than is allowed by the <foreign> element. The <distinct> element is provided for this purpose. Its attributes allow for additional information characterizing the nature of the linguistic distinction to be made in two distinct ways: the type attribute simply assigns a user-defined code of some kind to the word or phrase which assigns it to some register, sub-language, etc. No recommendations as to the set of values for this attribute are provided at this time, as little consensus exists in the field.

Alternatively, the remaining three attributes may be used in combination to place a word or phrase on a three-dimensional scale sometimes used in descriptive linguistics.[3] The time attribute places a word *diachronically*, for example as archaic, old-fashioned, contemporary, futuristic, etc.; the space attribute places a word *diatopically*, that is, with respect to a geographical classification, for example as national, regional, international, etc.; the social attribute places a word *diastatically*, that is, with respect to a social classification, for example as technical, polite, impolite, restricted, etc. Again, no recommendations are made for the values of these attributes at this time; the encoder should provide a description of the scheme used in the appropriate section of the header (see section 5.3 ('The Encoding Description') on p. 93).

Examples:

```
Next morning a boy in that dormitory confided to his
bosom friend, a <distinct type=psSlang>fag</distinct> of
Macrea's, that there was trouble in their midst which
King <distinct type=archaic>would fain</distinct> keep
secret.
```

```
Next morning a boy in that dormitory confided to his
bosom friend, a
<distinct time=1900 social=publicschool space=GB>fag</distinct>
of Macrea's, that there was trouble in their midst which
King <distinct time=archaic>would fain</distinct> keep
secret.
```

Where more complex (or more rigorous) interpretive analyses of the associations of a word are required, the more detailed and general mechanisms described in chapter 16 ('Feature Structures') on p. 397 should be preferred to these simple characterizations. It may also be preferable to record the kinds of analysis suggested here by means of the simple annotation

[2] The Oxford English Dictionary documents the phrase 'to come down' in the sense 'to bring or put down; *esp.* to lay down money; to make a disbursement' as being in use, mostly in colloquial or humorous contexts, from at least 1700 to the latter half of the 19th century.

[3] See, for example, *Sociolinguistics / Soziolinguistik (An international handbook of the science of language and society. Ein internationales Handbuch zur Wissenschaft von Sprache und Gesellschaft)* (Berlin, New York: De Gruyter, 1988), I, pp. 271 and 274.

element <note> described in section 6.8 ('Notes, Annotation, and Indexing') on p. 152, or the element described in section 15.3 ('Spans and Interpretations') on p. 387.

The <distinct> element has the following formal definition:

```
<!-- 6.3.2.3:  Highlighted phrases (cont'd)                    -->
<!-- (continuation of sec. 6.3.2.1)                            -->
<!ELEMENT distinct      - -  (%phrase.seq;)                       >
<!ATTLIST distinct           %a.global;
          type               CDATA               #IMPLIED
          time               CDATA               #IMPLIED
          space              CDATA               #IMPLIED
          social             CDATA               #IMPLIED       >
```

6.3.3 Quotation

This section discusses the following elements, all of which are often rendered by the use of quotation marks:

<q> contains a quotation or apparent quotation — a representation of speech or thought marked as being quoted from someone else (whether in fact quoted or not); in narrative, the words are usually those of of a character or speaker; in dictionaries, <q> may be used to mark real or contrived examples of usage. Attributes include:

- **who** identifies the speaker of a piece of direct speech.
- **type** may be used to indicate whether the quoted matter is spoken or thought, or to characterize it more finely. Sample values include:
  - ***spoken*** representation of direct speech, usually marked by quotation marks.
  - ***thought*** representation of thought, e.g. internal monologue.
- **direct** may be used to indicate whether the quoted matter is regarded as direct or indirect speech. Sample values include:
  - ***y*** speech or thought is represented directly.
  - ***n*** speech or thought is represented indirectly, e.g. by use of a marked verbal aspect.
  - ***unspecified*** no claim is made.

<quote> contains a phrase or passage attributed by the narrator or author to some agency external to the text.

<cit> A quotation from some other document, together with a bibliographic reference to its source.

<soCalled> contains a word or phrase for which the author or narrator indicates a disclaiming of responsibility, for example by the use of scare quotes or italics.

One form of presentational variation found particularly frequently in written and printed texts is the use of quotation marks. As with the typographic variations discussed in the preceding section, it is generally helpful to separate the encoding of the underlying textual feature (for example, a quotation or a piece of direct speech) from the encoding of its rendering (for example, the use of a particular style of quotation marks).

The most common and important use of quotation marks is, of course, to mark *quotation*, by which we mean simply any part of the text attributed by the author or narrator to some agency other than the narrative voice. Typical examples include passages cited from other works, for which the element <quote> may be used, and words or phrases attributed to other voices within the current work, for which the element <q> may be used. If this distinction between intra-textual and inter-textual voices cannot be made reliably, or is not of interest, then all quoted matter may simply be marked using the <q> tag. The editorial policy in this respect should be stated in the encoding description of the TEI Header. The <soCalled> element is used for cases where the author or narrator distances him or herself from the words in question without however attributing them to any other voice in particular.

Quotation may be rendered by changes in type face, by special punctuation marks (single or double or angled quotes, dashes, etc.) and by layout (indented paragraphs, etc.). If these characteristics are of interest, an appropriate value for the rend attribute should be given, to

record how the <q> or <quote> element is rendered. For discussion of suggested values for this attribute, see below.

Quotation marks themselves may, like other punctuation marks, be felt for some purposes to be worth retaining within a text, quite independently of their description by the rend attribute. Where this is done, an appropriate entity reference should be chosen from the standard entity sets listed in chapter 37 ('Obtaining TEI WSDs') on p. 985; this has the advantage that the entity may be redefined as null when the punctuation is to be ignored for some analytic purpose. Well-known ambiguities, such as whether the character ' represents an apostrophe or a closing single quotation mark, or whether the character " represents an opening or closing double quotation mark may all be resolved by the use of appropriate entity references, as discussed in section 6.2 ('Treatment of Punctuation') on p. 121.

Alternatively, the encoder may suppress all quotation marks, possibly recording their form using the rend attribute. Where this is done, the following list of entity names (taken from the public entity sets *ISOpub* and *ISOnum*) may be found useful to describe quotation-mark styles common in European and American typesetting:

ldquo double inverted comma (shaped like 66, superscript)
lsquo single inverted comma (shaped like 6, superscript)
rdquo double apostrophe (shaped like 99, superscript)
rsquo single apostrophe (shaped like 9, superscript)
ldquor double comma (shaped like 99, printed on base line)
lsquor single comma (shaped like 9, printed on base line)
laquo double guillemet open to the right
lsaquo single guillemet open to the right
raquo double guillemet open to the left
rsaquo single guillemet open to the left
mdash dash the width of a lowercase 'm'

These may be used in the rend attribute to show how the quotation was opened and closed. For example, if the words 'pre' and 'post' are used to indicate preceding and following punctuation, then the following example would describe a conventional American book printed using single quotation marks:

```
<q rend='PRE lsquo POST rsquo'>Who-e debel you?</q>
&mdash he at last said &mdash <q
rend='PRE lsquo POST rsquo'>you no speak-e,
damme, I kill-e.</q>  And so saying,
the lighted tomahawk began flourishing
about me in the dark.
```

The following example demonstrates alternative policies which may be adopted with respect to encoding of the punctuation used to mark quotation:

```
Adolphe se tourna vers lui :
<q>&mdash Alors, Albert, quoi de neuf?</q>
<q>&mdash Pas grand-chose.</q>
<q>&mdash Il fait beau,</q> dit Robert.


Adolphe se tourna vers lui :
<q rend='PRE mdash'>Alors, Albert, quoi de neuf ?</q>
<q rend='PRE mdash'>Pas grand-chose.</q>
<q rend='PRE mdash'>Il fait beau,</q> dit Robert.
```

To make explicit who is speaking, which is not always stated in the above example, the who attribute should be used:

```
Adolphe se tourna vers lui :
<q who='Adolphe'>&mdash Alors, Albert, quoi de neuf?</q>
<q who='Albert'>&mdash Pas grand-chose.</q>
<q who='Robert'>&mdash Il fait beau,</q> dit Robert.
```

The who attribute is also useful as a means of supplying a normalized form of the speaker's name, to facilitate selection of text by particular speakers. As indicated above, it may be supplied whether or not an indication of the speaker is given explicitly in the text.

Where investigation of "narrative voice" is the primary object of the encoding, it may be convenient to identify each speaker as a *participant* in the work, and to associate individual speeches with them by means of SGML's ID/IDREF mechanism. See section 23.2.2 ('The Participants Description') on p. 545 for discussion of the *participant description* component of the TEI Header.

For such analyses, it may also be useful to distinguish representations of speech from representations of thought, in modern printed texts often indicated by a change of typeface. The type attribute should be used for this purpose, as in this example:

```
<q type=speech>Oh yes,</q> said Henry, <q type=speech>I mean
Gordon Macrae, for example...</q>
<q type=thought>Jungian Analyst with Winebox! That's what you
called him, you callous bastard, didn't you? Eh? Eh?</q>
```

Quoted matter may be embedded within quoted matter, as when one speaker reports the speech of another:

```
<q who=Wilson>Spaulding, he came down into the office just this
day eight weeks with this very paper in his hand, and he
says:&mdash;<q who=Spaulding>I wish to the Lord, Mr. Wilson, that
I was a red-headed man.</q></q>
```

Direct speech nested in this way is treated in the same way as elsewhere: a change of rendition may occur, but the same element should be used. An encoder may however choose to distinguish between direct speech which contains quotations from extra-textual matter and direct speech itself, as in the following example:

```
<p><q>The Lord! The Lord! It is Sakya Muni himself,</q> the
lama half sobbed; and under his breath began the wonderful
Buddhist invocation:-
<q><quote><l>To Him the Way -- the Law -- Apart --
<l>Whom Maya held beneath her heart
<l>Ananda's Lord -- the Bodhisat
</quote>
And He is here! The Most Excellent Law is here also.  My
pilgrimage is well begun.  And what work! What work!</q>
```

Quotations from other works are often accompanied by a reference to their source. The <cit> element may be used to group together the quotation and its associated bibliographic reference, which should be encoded using the elements for bibliographic references discussed in section 6.10 ('Bibliographic Citations and References') on p. 162, as in the following example.

```
<div id=MM01 type=chapter><head>Chapter 1</head>
<epigraph>
  <cit><quote><l>Since I can do no good because a woman
  <l>Reach constantly at something that is near it.</quote>
  <bibl><title>The Maid's Tragedy</title>
        <author>Beaumont and Fletcher</author></bibl>
  </cit>
</epigraph>
<p>Miss Brooke had that kind of beauty which seems to be thrown into
relief by poor dress...
```

Like other bibliographic references, the citation attached to a quotation may be represented simply by a pointer, as in this example:

```
Lexicography has shown little sign of being affected by the
work of followers of J.R. Firth, probably best summarized
in his slogan, <cit><quote>You shall know a word by the company it
keeps.</quote> <ref target=FI57>(Firth, 1957)</ref></cit>
```

Unlike most of the other elements discussed in this chapter, direct speech and quotations may frequently contain other high-level elements such as paragraphs or verse lines, as well as

being themselves contained by such elements. Three possible solutions exist for this well-known structural problem:

- the quotation is broken into segments, each of which is entirely contained within a paragraph
- the quotation is marked up as part of a concurrent but independent hierarchy
- the quotation boundaries are represented by empty milestone tags

For further discussion, and several examples, see chapter 31 ('Multiple Hierarchies') on p. 633.

Finally, in this section, the element **<soCalled>** is provided for all cases in which quotation marks are used to distance the quoted text from the narrator or speaker. Common examples include the "scare" quotes often found in newspaper headlines and advertising copy, where the effect is to cast doubts on the veracity of an assertion:

```
<head>PM dodges <soCalled>election threat</soCalled>
in interview</head>
```

The same element should be used to mark a variety of special ironic usages. Some further examples follow:

```
He hated <soCalled>good</soCalled> books.

<soCalled>Croissants</soCalled> indeed! toast not good enough for you?
Although Chomsky's decision that all NL sentences are finite
objects was never justified by arguments from the attested
properties of NLs, it did have a certain
<soCalled>social</soCalled> justification.  It was
commonly assumed in works on logic until fairly recently
that the notion <mentioned>language</mentioned> is
necessarily restricted to finite strings.
```

The elements discussed in this section are formally defined as follows:

```
<!-- 6.3.3:  Highlighted phrases (cont'd)                    -->
<!-- (continuation of sec. 6.3.2.1)                          -->
<!ELEMENT q             - -  (%specialPara)                   >
<!ATTLIST q                  %a.global;
          type               CDATA              #IMPLIED
          direct             (y | n | unspecified)
                                                unspecified
          who                CDATA              #IMPLIED      >
<!ELEMENT quote         - -  (%specialPara;)                  >
<!ATTLIST quote              %a.global;                       >
<!ELEMENT cit           - -  ((q | quote) & (%m.bibl; |
                             %m.loc;))                        >
<!ATTLIST cit                %a.global;                       >
<!ELEMENT soCalled      - -  (%phrase.seq;)                   >
<!ATTLIST soCalled           %a.global;                       >
```

6.3.4 Terms, Glosses, and Cited Words

This section describes the following textual elements, all of which have in common that they may be variously realized using italics, quotation marks or other devices:

<term> contains a single-word, multi-word or symbolic designation which is regarded as a technical term.

<gloss> identifies a phrase or word used to provide a gloss or definition for some other word or phrase.

<mentioned> marks words or phrases mentioned, not used.

Technical terms are often italicized or emboldened upon first mention in printed texts; an explanation or gloss is sometimes given in quotation marks. Linguistic analyses conventionally cite words in languages under discussion in italics, providing a gloss immediately following marked with single quotation marks. Other texts in which individual words or phrases are *mentioned* (for example, as examples) rather than *used* may mark them either with italics or with quotation marks, and will gloss them less regularly.

A <term> may appear with or without a gloss, as may a <mentioned> element. Where the <gloss> is present, it may be linked to the term it is glossing by means of SGML's ID/IDREF mechanism. To establish such a link, the encoder should give an id value to the <term> or <mentioned> element and provide that id as the value of the target attribute on the <gloss> element. The following examples demonstrate this facility: for more discussion of this and other kinds of linkage within TEI documents, see chapter 14 ('Linking, Segmentation, and Alignment') on p. 331.

Examples:

```
We may define <term rend=sc id=tdpv>discoursal point of view</term>
as <gloss target=tdpv>the relationship, expressed through discourse
structure, between the implied author or some other addresser,
and the fiction.</gloss>
```

```
<gloss target=T1 rend=unmarked>A computational device
that infers structure from grammatical strings of words</gloss>
is known as a <term id=T1>parser</term>, and much of the
history of NLP over the last 20 years has been occupied with
the design of parsers.
```

```
There is thus a striking accentual difference between a
verbal form like <mentioned lang=grc
id=cw234>eluthemen</mentioned> <gloss target=cw234>we
were released,</gloss> accented on the second
syllable of the word, and its participial derivative
<mentioned id=cw235 lang=grc>lutheis</mentioned>
<gloss target=cw235>released,</gloss> accented on the last.
```

The elements discussed in this section have the following formal definitions:

```
<!-- 6.3.4:  Highlighted phrases (cont'd)                    -->
<!-- (continuation of sec. 6.3.2.1)                          -->
<!ELEMENT term          - -  (%phrase.seq;)                   >
<!ATTLIST term               %a.global;
          type               CDATA              #IMPLIED       >
<!ELEMENT mentioned     - -  (%phrase.seq;)                   >
<!ATTLIST mentioned          %a.global;                       >
<!ELEMENT gloss         - -  (%phrase.seq;)                   >
<!ATTLIST gloss              %a.global;
          target             IDREF              #IMPLIED       >
```

6.3.5 Some Further Examples

As a simple example of the elements discussed here, consider the following sentence:

"On the one hand the *Nibelungenlied* is associated with the new rise of romance of twelfth-century France, the *romans d'antiquité*, the romances of Chrétien de Troyes, and the German adaptations of these works by Heinrich van Veldeke, Hartmann von Aue, and Wolfram von Eschenbach." A first approximation to the encoding of this sentence, might be simply to record the fact that the phrases printed above in italics are highlighted, as follows:

```
On the one hand the <hi rend=italic>Nibelungenlied</hi> is
associated with the new rise of romance of twelfth-century
France, the <hi rend=italic lang=fr>romans
d'antiquit&eacute;</hi>, the romances of Chr&eacute;tien de
Troyes, ...
```

This encoding would however lose the important distinction between an italicized title and an italicized foreign phrase. Many other phrases might also be italicized in the text, and a retrieval program seeking to identify foreign terms (for example) would not be able to produce reliable results by simply looking for italicized words. Where economic and intellectual constraints permit, therefore, it would be preferable to encode both the function of the highlighted phrases and their appearance, as follows:

```
On the one hand the <title
rend=italic>Nibelungenlied</title> is associated with the
```

```
new rise of romance of twelfth-century France, the <foreign
rend=italic>romans d'antiquit&eacute;</foreign>, the
romances of Chr&eacute;tien de Troyes, ...
```

In this example, the decision as to which textual features are distinguished by the highlighting is relatively uncontroversial. As a less straightforward example, consider the use of italic font in the following passage from Samuel Richardson's *Clarissa* (1747). "A pretty common case, I believe; in all *vehement* debatings. She says I am *too witty*; Anglicé, *too pert*; I, that she is *too wise*; that is to say, being likewise put into English, *not so young as she has been*: in short, she is grown so much into a *mother*, that she had forgotten she ever was a *daughter*. ..."

Clearly, the word 'vehement' is not italicized for the same reason as the phrase 'not so young as she has been'; the former is emphasized, while the latter is proverbial. It also provides an ironic gloss for the words 'too wise', in the same way as 'too pert' glosses 'too witty'. The glossed phrases are not however technical terms or cited words, but quoted phrases, as if Clarissa were putting words into her own and her mother's mouths. Finally, the words 'mother' and 'daughter' are apparently italicized simply to oppose them in the sentence; certainly they do not fit into any of the categories so far proposed as reasons for italicizing. Note also that the word 'Anglicé' is not italicized although it is not generally considered an English word.

The following sample encoding for the above passage attempts to take into account all the above points:

```
A pretty common case, I believe; in all
<emph>vehement</emph> debatings.  She says I am
<q rend=italic>too witty</q>; <foreign lang=la
rend=roman>Anglic&egrave;</foreign>, <gloss rend=italic>too
pert</gloss>; I, that she is <q rend=italic>too wise</q>;
that is to say, being likewise put into English, <gloss
rend=italic>not so young as she has been</gloss>:  in short,
she is grown so much into a <hi rend=italic>mother</hi>,
that she had forgotten she ever was a
<hi rend=italic>daughter</hi>.
```

6.4 Names, Numbers, Dates, Abbreviations, and Addresses

This section describes a number of textual features which it is often convenient to distinguish from their surrounding text. Names, dates and numbers are likely to be of particular importance to the scholar treating a text as source for a database; distinguishing such items from the surrounding text is however equally important to the scholar primarily interested in lexis.

The treatment of these textual features proposed here is not intended to be exhaustive: fuller treatments for names, numbers, measures and dates are provided in the additional tag set for names and dates (see chapter 20 ('Names and Dates') on p. 487).

6.4.1 Referring Strings

A *referring string* is a phrase which refers to some person, place, object etc. Two elements are provided to mark such strings:

<rs> contains a general purpose name or referring string. Attributes include:

type indicates more specifically the object referred to by the referencing string. Values might include "person", "place", "ship", "element" etc.

<name> contains a proper noun or noun phrase. Attributes include:

type indicates the type of the object which is being named by the phrase.

Where it is thought useful to do so, the kind of object referred to may be specified using the **type** attribute.

Examples include:

```
<q>My dear <rs type=person>Mr. Bennet</rs>, </q>
said his lady to him one day, <q>have you heard
that <rs type=place>Netherfield Park</rs> is let
at last?</q>

Collectors of water-rents were appointed by the
<rs type=organization>Watering Committee</rs>.
They were paid a commission not exceeding four per
cent, and gave bond.


It being one of the principles of the
<rs type=org>Circumlocution Office</rs> never, on any
account whatsoever, to give a straightforward answer,
<rs type=person>Mr Barnacle</rs> said, <q>Possibly.</q>
```

As the following example shows, the <rs> element may be used for any reference to a person, place, etc., not only to references in the form of a proper noun or noun phrase.

```
<q>My dear <rs type=person>Mr. Bennet</rs>,</q>
said <rs type=person>his lady</rs> to him
one day...
```

The <name> element by contrast is provided for the special case of referencing strings which consist only of proper nouns; it may be used synonymously with the <rs> element, or nested within it if a referring string contains a mixture of common and proper nouns. The following example shows an alternative way of encoding the short sentence from *Pride and Prejudice* quoted above:

```
<q>My dear <name type=person>Mr. Bennet</name>,</q>
said <rs type=person>his lady</rs> to him one day,
<q>have you heard that <name type=place>Netherfield
Park</name> is let at last?</q>
```

The following example shows how a proper name may be nested within a referring string:

```
<rs>His Excellency the Life President,
<name>Ngwazi Dr H. Kamuzu Banda</name></rs>
<!-- ... -->
```

Simply tagging something as a name is generally not enough to enable automatic processing of personal names into the canonical forms usually required for reference purposes. The name as it appears in the text may be inconsistently spelled, partial, or vague. Moreover, name prefixes such as 'van' or 'de la', may or may not be included as part of the reference form of a name, depending on the language and country of origin of the bearer.

The following attributes, common to all members of the *names* element class, are provided to help overcome these difficulties:

key provides an alternative identifier for the object being named, such as a database record key.
reg gives a normalized or regularized form of the name used.

Either or both of these attributes may be specified, as appropriate. The key attribute may be useful as a means of gathering together all references to the same individual or location scattered throughout a document:

```
<q>My dear <rs type=person key=BENM1>Mr. Bennet</rs>,
</q> said <rs type=person key=BENM2>his lady</rs>
to him one day, <q>have you heard that
<rs type=place key=NETP1>Netherfield Park</rs>
is let at last?</q>
```

This use should be distinguished from the case of the reg (regularization) attribute, which provides a means of marking the standard form of a referencing string as demonstrated below:

```
My personal life during the administration of
<rs type=person key=POJA1 reg='Polk, James K.'>Col. Polk</rs>
has but poorly compensated me for the suspended
enjoyments and pursuits of private and professional spheres
```

```
<name type=person key=VOM1 reg='Volanges, Mme de'>
Mme. de Volanges</name>
marie sa fille: c'est encore un secret; mais elle m'en
a fait part hier.
<name type=person key=WADLM1 reg='de la Mare, Walter'>
Walter de la Mare
</name>
was born at
<name key=Ch1 type=place>Charlton</name>, in
<name key=KT1 type=county>Kent</name>, in 1873.
<name type=place>Montaillou</name> is not a large parish.
At the time of the events which led to
<name type=person reg='Benedict XII, Pope of Avignon
(Jacques Fournier)'>Fournier's</name> investigations,
the local population consisted of between 200 and
250 inhabitants.
```

This method is adequate for many simple applications. For more complex applications, such as onomastics, or wherever a detailed analysis of the component parts of a name is needed, the specialized elements described in chapter 20 ('Names and Dates') on p. 487 or the analytical tools described in chapter 16 ('Feature Structures') on p. 397 should be used.

These elements are formally declared as follows:

```
<!-- 6.4.1:  Proper Nouns                                    -->
<!ELEMENT name          - -  (%phrase.seq;)                    >
<!ATTLIST name               %a.global;
                             %a.names;
          type               CDATA              #IMPLIED       >
<!ELEMENT rs            - -  (%phrase.seq)                     >
<!ATTLIST rs                 %a.global;
                             %a.names;
          type               CDATA              #IMPLIED       >
<!-- This fragment is used in sec. 6.12                        -->
```

6.4.2 Addresses

The simplest way of encoding an address is to regard it as a series of distinct lines, just as they might be printed on an envelope. The following elements support this view:

<address> contains a postal or other address, for example of a publisher, an organization, or an individual.

<addrLine> contains one line of a postal or other address.

Alternatively, an address may be encoded as a structure composed of the following elements, which constitute the *addrPart* element class:

<street> a full street address including any name or number identifying a building as well as the name of the street or route on which it is located.

<name> contains a proper noun or noun phrase. Attributes include:

type indicates the type of the object which is being named by the phrase.

<postCode> contains a numerical or alphanumeric code used as part of a postal address to simplify sorting or delivery of mail.

<postBox> contains a number or other identifier for some postal delivery point other than a street address.

Any number of *addrPart* elements may appear within an address and in any order. None of them is required. Where code letters are commonly used in addresses (for example, to identify regions or countries) a useful practice is to supply the full name of the region or country as the content of the element, but to supply the abbreviatory code as the value of the global n attribute, so that (for example) an application preparing formatted labels can readily find the required information. Other components of addresses should be represented using the general-purpose <name> element.

Some examples follow:

```
<address>
<addrLine>110 Southmoor Road, </>
<addrLine>Oxford, OX2 6RB,</>
<addrLine>UK</>
</address>
```

The above address could also be represented as follows :

```
<address>
<street>110 Southmoor Road</street>
<name type=city n=OX>Oxford</name>
<postCode>OX2 6RB</postCode>
<name type=country n=UK>United Kingdom</name>
</address>
```

The order of elements within an address is highly culture-specific, and is therefore unconstrained:

```
<address>
<name type=org>Universit&agrave; di Bologna
<name type=country n=I>Italy</><postCode>40126
<name type=city>Bologna
<street>via Marsala 24
</address>
```

For further discussion of ways of regularizing the names of places, see section 6.4 ('Names, Numbers, Dates, Abbreviations, and Addresses') on p. 132. A full postal address may also include the name of the addressee, tagged as above using the general purpose **<name>** element. When the additional tag set for names and dates is enabled, more specific elements such as **<publisher>** or **<org>** may be used, as further discussed in chapter 20 ('Names and Dates') on p. 487.

The **<address>** element and its components are formally described as follows:

```
<!-- 6.4.2:  Addresses and their components                    -->
<!ELEMENT address       - O  (addrLine+ | (%m.addrPart)*)       >
<!ATTLIST address           %a.global;                          >
<!ELEMENT addrLine      - o  (%phrase.seq)                      >
<!ATTLIST addrLine          %a.global;                          >
<!ELEMENT street        - o  (%phrase.seq)                      >
<!ATTLIST street            %a.global;                          >
<!ELEMENT postCode      - o  (#PCDATA)                          >
<!ATTLIST postCode          %a.global;                          >
<!ELEMENT postBox       - o  (#PCDATA)                          >
<!ATTLIST postBox           %a.global;                          >
<!-- Other components of addresses should be represented      -->
<!-- using the general purpose NAME element                   -->

<!-- This fragment is used in sec. 6.12                       -->
```

6.4.3 Numbers and Measures

This section describes two elements provided for the simple encoding of numbers and measures and gives some indication of circumstances in which this may usefully be done. The following phrase level elements are provided for this purpose:

<num> contains a number, written in any form. Attributes include:

type indicates the type of numeric value. Suggested values include:

cardinal absolute number, e.g. "21", "21.5"

ordinal ordinal number, e.g. "21st"

fraction fraction, e.g. "one half" or "three halves"

percentage e.g. "ten percent"

value supplies the value of the number in an application-dependent standard form.

<measure> contains a word or phrase referring to some quantity of an object or commodity, usually comprising a number, a unit and a commodity name. Attributes include:

type specifies the type of unit in which the measure is expressed. Sample values include:

weight measure of weight e.g. kg, pound.

count unit of count, e.g. dozen, score.

length measure of length, e.g. pole, mm.

area measure of area e.g. acre, hectare.

volume measure of volume e.g. litre, gallon.

currency unit of currency e.g. ecu, escudo, mark.

Like names or abbreviations, numbers can occur virtually anywhere in a text. Numbers are special in that they can be written with either letters or digits ('twenty-one', 'xxi', and '21') and their presentation is language-dependent (e.g. English '5th' becomes Greek '5.'; English '123,456.78' equals French '123.456,78').

For many kinds of application, e.g. natural-language processing or machine translation, numbers are not regarded as "lexical" in the same way as other parts of a text. For these and other applications, the <num> element provides a convenient method of distinguishing numbers from the surrounding text. For other kinds of application, numbers are only useful if normalized: here the <num> element is useful precisely because it provides a standardized way of representing a numerical value.

For example:

```
<num value='33'>xxxiii</num>
<num type=cardinal value='21'>twenty-one</num>
<num type=percentage value='10'>ten percent</num>
<num type=percentage value='10'>10%</num>
<num type=ordinal value='5'>5th</num>

<num type=fraction value='0,5'>one half</num>
<num type=fraction value='0,5'>1/2</num>
```

The word 'measure' is used here to refer to a special kind of referring string, the referent of which is a "virtual object". In its fullest form, a measure consists of a number, a phrase expressing units of measure and a phrase expressing the commodity being measured. Not all of these components need be present in every case. For some applications, particularly quantitative ones, the internal components of measure need to be marked so that their values can be calculated. Thus, in order to evaluate a monetary measure according to some standard, it is necessary to mark its currency unit (e.g. US dollars, pounds sterling). Similarly, the expression '2 ounces' will have a different meaning when it is associated with 'flour' from that which it has when associated with 'water'.

Such applications will require the elements discussed in chapter 20 ('Names and Dates') on p. 487, or the more powerful analytical tools discussed in chapter 16 ('Feature Structures') on p. 397. Elsewhere, it may be sufficient simply to encode measures as such, perhaps also indicating their numeric content with the <num> element, as in the following examples:

```
<l>I've measured it from side to side
<l>'Tis
<measure type=length reg='0.924m'>
    <num value=3>three</num>
    feet
</measure>
long, and
<measure type=length reg='0.616m'>
   <num value=2>two</num>
   feet
</measure>
wide.
</l>
```

As the above example also demonstrates, the <measure> element is a member of the class *names* like other referencing strings, and may thus bear a reg attribute to indicate a normalized

value. It may also carry a key attribute to indicate a database key value as in the following example:

```
<list>
<item><measure key=BH2 type=volume>
         <num value=2>ii</num>
          bags hops
       </measure>
<item> <measure key=TW6 type=volume>
         <num value=6>six</num>
        trusses Woolen and linen goods
       </measure>
<item><measure  key=WC5 type=weight>
        5 tonnes coale
      </measure>
<!-- ... -->
</list>
```

These elements are formally defined as follows:

```
<!-- 6.4.3:  Numbers and measures                              -->
<!ELEMENT num           - -  (%phrase.seq;)                     >
<!ATTLIST num                %a.global;
          type               CDATA               #IMPLIED
          value              CDATA               #IMPLIED      >
<!ELEMENT measure       - -  (%phrase.seq;)                     >
<!ATTLIST measure            %a.global;
                             %a.names;
          type               CDATA               #IMPLIED      >
<!-- This fragment is used in sec. 6.12                        -->
```

6.4.4 Dates and times

Dates and times, like numbers, can appear in widely varying culture- and language-dependent forms, and can pose similar problems in automatic language processing. The following elements are provided to identify them:

<date> contains a date in any format. Attributes include:

- **calendar** indicates the system or calendar to which the date belongs.
- **value** gives the value of the date in some standard form, usually yyyy-mm-dd.

<time> contains a phrase defining a time of day in any format. Attributes include:

- **value** gives the value of the time in a standard form.

<dateRange> contains two dates or another phrase delimiting a time period. Attributes include:

- **calendar** indicates the system or calendar to which the date belongs.
- **from** indicates the starting point of the period in standard form.
- **to** indicates the ending point of the period in standard form.
- **exact** indicates the precision to be attached to either or both dates specified. Sample values include:
  - ***to*** the to date is exact
  - ***from*** the from date is exact
  - ***both*** both dates are exact
  - ***none*** both dates are approximate or unspecified

<timeRange> contains two times or another phrase indicating a time period. Attributes include:

- **from** indicates the starting point of the time period in standard form.
- **to** indicates the ending point of the time period in standard form.
- **exact** indicates the precision to be attached to either or both times specified. Sample values include:

to the to time is exact
from the from time is exact
both both times are exact
none both times are approximate or unspecified

Dates can occur virtually anywhere in a text, but in some contexts (e.g. bibliographic citations) their encoding is recommended or required rather than optional. Times can also appear anywhere but are generally optional.

Partial dates or times (e.g. '1990', 'September 1990', 'twelvish') can be expressed in the value attribute by simply omitting a part of the value supplied. Imprecise dates or times (for example 'early August', 'some time after ten and before twelve') may be expressed as date or time ranges. If either end of the date or time range is known to be accurate (for example, 'at some time before 1230', 'a few days after Hallowe'en'), the exact attribute may be used to specify this.

Where the certainty (i.e. reliability) of the date or time itself is in question, rather than its precision, the encoder should record this fact using the mechanisms discussed in chapter 17 ('Certainty and Responsibility') on p. 435.

These mechanisms are useful primarily for fully specified dates or times known with certainty. If component parts of dates or times are to be marked up, or if a more complex analysis of the meaning of a temporal expression is required, the techniques described in chapter 20 ('Names and Dates') on p. 487 should be used in preference to the simple method outlined here.

Examples:

```
<date value='1980-02-21'>21 Feb 1980</date>
```

```
Given on the <date value='1977-06-12'>Twelfth Day of June
in the Year of Our Lord One Thousand Nine Hundred and
Seventy-seven of the Republic the Two Hundredth and first
and of the University the Eighty-Sixth.</date>
```

```
<date value='1990'>1990</date>
<date value='1990-09'>September 1990</date>
```

```
Those five years &mdash; <dateRange from=1918 to=1923>
1918 to 1923</dateRange> &mdash had been, he suspected,
somehow very important.
```

```
The Eddic poems are preserved in a unique
manuscript (Codex Regius 2365) from
<dateRange from=1250 to=1300>
the second half of the thirteenth
century</dateRange>, and <title>Hervarar
saga</title> dates from <date value=1300>
around 1300</date>.
```

These elements are formally defined as follows:

```
<!-- 6.4.4:  Dates and times                                    -->
<!ELEMENT date          - -  (%phrase.seq;)                      >
<!ATTLIST date               %a.global;
          calendar           CDATA              #IMPLIED
          value              CDATA              #IMPLIED
          certainty          CDATA              #IMPLIED       >
<!ELEMENT dateRange     - O  (%phrase.seq;)                      >
<!ATTLIST dateRange          %a.global;
          calendar           CDATA              #IMPLIED
          from               CDATA              #IMPLIED
          to                 CDATA              #IMPLIED
          exact              (to | from | both | none)
                                                #IMPLIED       >
<!ELEMENT time          - -  (%phrase.seq;)                      >
<!ATTLIST time               %a.global;
          value              CDATA              #IMPLIED
          type               (am | pm | 24hour | descriptive)
                                                #IMPLIED
```

```
            zone                CDATA                #IMPLIED       >
  <!ELEMENT timeRange      - -  (%phrase.seq;)                      >
  <!ATTLIST timeRange           %a.global;
            from                CDATA                #IMPLIED
            to                  CDATA                #IMPLIED
            exact               (to | from | both | none)
                                                     #IMPLIED       >
  <!-- This fragment is used in sec. 6.12                          -->
```

6.4.5 Abbreviations and Their Expansions

It is sometimes desirable to mark abbreviations in the copy text, whether to trigger special processing for them, to provide the full form of the word or phrase abbreviated, or to allow for different possible expansions of the abbreviation. Abbreviations may be transcribed as they stand, or expanded; they may be left unmarked, or marked using these tags:

<abbr> contains an abbreviation of any sort. Attributes include:

- **expan** gives an expansion of the abbreviation.
- **resp** signifies the editor or transcriber responsible for supplying the expansion of the abbreviation held as the value of the expan attribute.
- **type** allows the encoder to classify the abbreviation according to some convenient typology. Sample values include:
  - ***suspension*** the abbreviation provides the first letter(s) of the word or phrase, omitting the remainder.
  - ***contraction*** the abbreviation omits some letter(s) in the middle.
  - ***brevigraph*** the abbreviation comprises a special symbol or mark.
  - ***superscription*** the abbreviation includes writing above the line.
  - ***acronym*** the abbreviation comprises the initial letters of the words of a phrase.
  - ***title*** the abbreviation is for a title of address (Dr, Ms, Mr, ...)
  - ***organization*** the abbreviation is for the name of an organization.
  - ***geographic*** the abbreviation is for a geographic name.
- **cert** signifies the degree of certainty ascribed to the expansion of the abbreviation.

<expan> contains the expansion of an abbreviation. Attributes include:

- **abbr** gives the abbreviation in its unexpanded form.
- **resp** signifies the editor or transcriber responsible for supplying the expansion of the abbreviation held as the content of the <expan> element.
- **type** allows the encoder to classify the abbreviation according to some convenient typology.
- **cert** signifies the degree of certainty ascribed to the expansion of the abbreviation.

The <abbr> element is useful as a means of distinguishing semi-lexical items such as acronyms or jargon:

```
We can sum up the above discussion as follows:  the identity of a
<abbr>CC</abbr> is defined by that calibration of values which
motivates the elements of its <abbr>GSP</abbr>; ...
```

```
Every manufacturer of <abbr>3GL</abbr> or <abbr>4GL</abbr>
languages is currently nailing on <abbr>OOP</abbr> extensions.
```

The type attribute may be used to distinguish types of abbreviation by their function, and the expan attribute may be used to supply an expansion:

```
<abbr type=title>Dr.</abbr> <abbr type=initial>M.</abbr> Deegan
is the Director of the <abbr expan='Computers in Teaching Initiative'
type=acronym>CTI</abbr> Centre for Textual Studies.
```

Abbreviations such as 'Dr M' above may be treated as two abbreviations, as above, or as one:

```
<abbr>Dr. M.</abbr> Deegan is the Director of the
<abbr>CTI</abbr> Centre for Textual Studies.
```

This element is particularly useful where manuscript materials in which abbreviation is very frequent are being transcribed. For example:

```
<l>Ex<abbr type=brevigraph expan='per'
          Resp=PG>&per;</abbr>ience, thogh noon auctoritee
<l>Were in this world, is right ynogh for me
<l>To speke of wo that is in mariage;
```

Here an entity reference *per* has been used to represent the common manuscript symbol 'crossed-p', and its expansion supplied in the associated <abbr> tag. The same lines might be transcribed, expanded, as follows:

```
<l>Ex<expan type=brevigraph abbrev='&per;'
         Resp=PG>per</expan>ience, thogh noon auctoritee
<l>Were in this world, is right ynogh for me
<l>To speke of wo that is in mariage;
```

In practice, it may be most convenient to transcribe the abbreviation as an entity reference; this allows the entity reference itself to be expanded either as an <abbr> or as an <expan> element, depending on the processing to be done at the moment. (For further discussion of such documentation, see section 25.4.3 ('Documenting Coded Character Sets and Entity Sets') on p. 581.) The text shown here:

```
<l>Ex&per;ience, thogh noon auctoritee
<l>Were in this world, is right ynogh for me
<l>To speke of wo that is in mariage;
```

may be expanded as desired by providing the appropriate choice between the two entity declarations:

```
<!ENTITY per "<abbr type=brevigraph expan='per'
              Resp=PG>&p.crossed;</abbr>" >
<!ENTITY per "<expan type=brevigraph abbrev='&p.crossed;'
              Resp=PG>per</expan>" >
```

For further discussion of manuscript abbreviations, see chapter 18 ('Transcription of Primary Sources') on p. 443.

These elements are formally defined as follows:

```
<!-- 6.4.5:  Abbreviations                                    -->
<!ELEMENT abbr          - -  (%phrase.seq;)                    >
<!ATTLIST abbr               %a.global;
          expan              CDATA              #IMPLIED
          resp               IDREF              %INHERITED
          cert               CDATA              #IMPLIED
          type               CDATA              #IMPLIED       >
<!ELEMENT expan         - -  (%phrase.seq;)                    >
<!ATTLIST expan              %a.global;
          abbr               CDATA              #IMPLIED
          resp               IDREF              %INHERITED
          cert               CDATA              #IMPLIED
          type               CDATA              #IMPLIED       >
<!-- This fragment is used in sec. 6.12                        -->
```

6.5 Simple Editorial Changes

As in editing a printed text, so in encoding a text in electronic form, it may be necessary to accommodate editorial comment on the text and to render account of any changes made to the text in preparing it. The tags described in this section may be used to record such editorial interventions, whether made by the encoder, by the editor of a printed edition used as a copy text, by earlier editors, or by the copyists of manuscripts.

The tags described here handle most common types of editorial intervention and stereotyped comment; where less structured commentary of other types is to be included, it should be marked using the <note> element described in section 6.8 ('Notes, Annotation, and Indexing') on p. 152. Systematic interpretive annotation is also possible using the various methods described in chapter

14 ('Linking, Segmentation, and Alignment') on p. 331. The examples given here illustrate only simple cases of editorial intervention; in particular, they permit economical encoding of two alternative readings of a text only. To encode more than two views of any one segment of text, the mechanisms described in chapters 14 ('Linking, Segmentation, and Alignment') on p. 331 and 19 ('Critical Apparatus') on p. 467 must be used.

The first two pairs of elements here discussed (<sic> and <corr>, <reg> and <orig>) may both be used to record simultaneously a text in its "original", uncorrected and unaltered form and also in an "edited" form. In this way they resemble the pair <abbr> and <expan>, described in section 6.4.5 ('Abbreviations and Their Expansions') on p. 139. Such paired elements enable software to move automatically from one "view" of the text to the other.

Three categories of editorial intervention are discussed in this section:

- correction (or non-correction) of apparent errors
- regularization (or non-regularization) of variant, irregular, non-standard, or eccentric forms
- editorial additions, suppressions, and omissions

A more extended treatment of the use of these tags in transcriptional and editorial work is given in chapter 18 ('Transcription of Primary Sources') on p. 443.

6.5.1 Correction of Apparent Errors

When the copy text is manifestly faulty, an encoder or transcriber may elect simply to correct it without comment. For scholarly purposes, it will often be more generally useful to record both the correction and the original state of the text. The elements described here enable this to be done is such a way as not to distract the reader.

<sic> contains text reproduced although apparently incorrect or inaccurate. Attributes include:

- **corr** gives a correction for the apparent error in the copy text.
- **resp** signifies the editor or transcriber responsible for suggesting the correction held as the value of the corr attribute.
- **cert** signifies the degree of certainty ascribed to the correction held as the value of the corr attribute.

<corr> contains the correct form of a passage apparently erroneous in the copy text. Attributes include:

- **sic** gives the original form of the apparent error in the copy text.
- **resp** signifies the editor or transcriber responsible for suggesting the correction held as the content of the <corr> element.
- **cert** signifies the degree of certainty ascribed to the correction held as the content of the <corr> element.

The following examples show alternative treatment of the same material. The copy text reads:

"Another property of computer-assisted historical research is that data modelling must permit any one textual feature or part of a textual feature to be a part of more than one information model and to allow the researcher to draw on several such models simultaneously, for example, to select from a machine-readable text those marginal comments which indicate that the date's mentioned in the main body of the text are incorrect."

An encoder may choose to correct the typographic error, either silently or with an indication that a correction has been made, as follows:

```
... marginal comments which indicate that the
<corr>dates</corr> mentioned in
the main body of the text are incorrect.
```

Alternatively, the encoder may simply record the typographic error without correcting it, either without comment or with a <sic> element to indicate the error is not a transcription error in the encoding:

```
... marginal comments which indicate that the
<sic>date's</sic> mentioned in
the main body of the text are incorrect.
```

If the encoder elects both to record the original source text and to provide a correction for the sake of word-search and other programs, either <sic> or <corr> may be used with the appropriate attribute:

```
... marginal comments which indicate that the
<sic corr='dates' resp=MSM>date's</sic> mentioned in
the main body of the text are incorrect.
... marginal comments which indicate that the
<corr sic="date's" resp='MSM'>dates</corr> mentioned in
the main body of the text are incorrect.
```

If both readings are given, the choice between <sic> and <corr> is largely a question of individual preference; since both record the same information, either may be mechanically transformed into the other. If the original reading contains SGML tags, it will prove more convenient to use <sic> than <corr> (and vice versa if there are tags within the corrected reading), since SGML tags are not recognized in attribute values. If both readings contain subordinate tags, then recourse must be had to the methods described in chapter 19 ('Critical Apparatus') on p. 467.

The cert attribute on the <sic> and <corr> elements permits a statement of the degree of editorial confidence in a particular correction. For example, using a confidence scale of one to ten, an editor may indicate the conjectural status of a correction by assigning a value to this attribute of less than ten. In the following instance, some uncertainty is expressed concerning a commonly-accepted emendation:

```
An <corr sic='Antony' cert=8>Autumn</corr> it was,
That grew the more by reaping
```

See further the discussion in section 18.1.3 ('Correction and Conjecture') on p. 447.

Where the correction takes the form of adding text, the encoder must choose whether to use the <corr> (or <sic>) tag, the <add> tag (see section 6.5.3 ('Additions, Deletions and Omissions') on p. 144 below), or the more detailed facilities provided by the additional tag set for primary source description. The following discussion may be helpful when making this decision:

- where the correction is an addition by a scribe or author in a manuscript or other primary source (typescript, proof or galley, etc.) then either <corr> (or <sic>) or <add> might be appropriate, depending on the circumstances. The <add> tag is more expressive, and may convey information on just how the addition was performed (hand, place, etc.) which the <corr> tag cannot. See further the discussion in section 18.1.5 ('Substitutions') on p. 452.
- where the correction is an addition by a transcriber or editor, correcting a perceived deficiency in the text but in circumstances where there is no clearly assertable reason for the deficiency (as a manuscript lacuna, or damage to the page) the <corr> tag should be used. The <add> tag should not be used in this case.
- where the correction is an addition by a transcriber or editor, correcting a perceived deficiency in the text and where there is a clearly assertable reason for the deficiency (as a manuscript lacuna, or damage to the page) which the encoder wishes to record, or when supplying text from a parallel version of the text, the <supplied> element provided by the additional tag set for primary source description should be used (see section 18.1.3 ('Correction and Conjecture') on p. 447).

The formal definition of these elements is as follows:

```
<!-- 6.5.1:  Editorial tags for correction                    -->
<!ELEMENT sic           - -  (%specialPara;)                     >
<!ATTLIST sic                %a.global;
          corr               CDATA              #IMPLIED
          resp               IDREF              %INHERITED
          cert               CDATA              #IMPLIED       >
<!ELEMENT corr          - -  (%specialPara;)                     >
<!ATTLIST corr               %a.global;
          sic                CDATA              #IMPLIED
          resp               CDATA              %INHERITED
          cert               CDATA              #IMPLIED       >
<!-- This fragment is used in sec. 6.12                         -->
```

6.5.2 Regularization and Normalization

When the source text makes extensive use of variant forms or non-standard spellings, it may be desirable for a number of reasons to *regularize* it: that is, to provide "standard" or "regularized" forms equivalent to the non-standard forms.[4]

As with other such changes to the copy text, the changes may be made silently (in which case the TEI header should specify the types of silent changes made) or may be explicitly marked using the following elements:

<reg> contains a reading which has been regularized or normalized in some sense. Attributes include:

- **orig** gives the unregularized form of the text as found in the source copy.
- **resp** identifies the individual responsible for the regularization of the word or phrase.

<orig> contains the original form of a reading, for which a regularized form is given in an attribute value. Attributes include:

- **reg** gives a regularized (normalized) form of the text.
- **resp** identifies the individual responsible for the regularization of the word or phrase.

Typical applications for these elements include the production of editions intended for student or lay readers, linguistic research in which spelling or usage variation is not the main question at issue, production of spelling dictionaries, etc.

Consider this 16th-century text: "how godly a dede it is to overthrowe so wicked a race the world may judge: for my part I thinke there canot be a greater sacryfice to God."

An encoder may choose to preserve the original spelling of this text, but simply flag it as nonstandard by using the <orig> element with no attributes specified, as follows:

```
how godly a <orig>dede</orig> it is to
<orig>overthrowe</orig> so wicked a race the
world may judge:  for my part I <orig>thinke</orig>
there <orig>canot</orig> be a greater
<orig>sacryfice</orig> to God.
```

Alternatively, the encoder may simply indicate that certain words have been modernized by using the <reg> element with no attributes specified, as follows:

```
how godly a <reg>deed</reg> it is to
<reg>overthrow</reg> so wicked a race the
world may judge:  for my part I <reg>think</reg>
there <reg>cannot</reg> be a greater
<reg>sacrifice</reg> to God.
```

More usefully, the encoder may elect to record both old and new spellings, so that (for example) the same electronic text may serve as the basis of an old- or new-spelling edition:

```
how godly a <reg orig='dede'>deed</reg> it is to
<reg orig='overthrowe'>overthrow</reg> so wicked a race the
world may judge:  for my part I <reg orig='thinke'>think</reg>
there <reg orig='canot'>cannot</reg> be a greater
<reg orig='sacryfice'>sacrifice</reg> to God.
```

Or the <orig> tag might be preferred

```
how godly a <orig reg='deed'>dede</orig> it is to
<orig reg='overthrow'>overthrowe</orig> so wicked a race the
world may judge:  for my part I <orig reg='think'>thinke</orig>
there <orig reg='cannot'>canot</orig> be a greater
<orig reg='sacrifice'>sacryfice</orig> to God.
```

The resp attribute should be used to specify the agency responsible for the regularization. This may be an identifiable individual, for example an editor, or a descriptive phrase such as 'copyist'. For example, in the first stanza of the Old Norse poem *Grógaldr*, the manuscript form 'dura' is usually regularized in modern editions to 'dyra' 'doors'. The manuscript's "vek ek þik dauðra dura" might thus be recorded together with its regularization in two ways, as follows:

[4] In some contexts, the term 'regularization' has a narrower and more specific significance than that proposed here: the <reg> element may be used for any kind of regularization, including normalization, standardization, and modernization.

```
vek ek &thorn;ik dau&eth;ra <reg orig='dura' resp=ed>dyra</reg>
```

or:

```
vek ek &thorn;ik dau&eth;ra <orig reg='dyra' resp=ed>dura</orig>
```

These elements are formally defined as follows:

```
<!-- 6.5.2:  Editorial tags for regularization                -->
<!ELEMENT reg           - -  (%phrase.seq;)                    >
<!ATTLIST reg                %a.global;
          orig               CDATA               #IMPLIED
          resp               CDATA               #IMPLIED      >
<!ELEMENT orig          - -  (%phrase.seq;)                    >
<!ATTLIST orig               %a.global;
          reg                CDATA               #IMPLIED
          resp               CDATA               #IMPLIED      >
<!-- This fragment is used in sec. 6.12                        -->
```

6.5.3 Additions, Deletions and Omissions

The following elements are used to indicate when words or phrases have been omitted from, added to, or marked for deletion from, a text. Like the other editorial elements, they allow for a wide range of editorial practices:

<gap> indicates a point where material has been omitted in a transcription, whether for editorial reasons described in the TEI header, as part of sampling practice, or because the material is illegible or inaudible. Attributes include:

- **desc** gives a description of the omitted text.
- **reason** gives the reason for omission. Sample values include "sampling", "illegible", "inaudible", "irrelevant", "canceled", "canceled and illegible".
- **extent** indicates approximately how much text has been omitted from the transcription, in letters, minims, inches, or any appropriate unit, either because of editorial policy or because a deletion, damage or other cause has rendered transcription impossible.
- **resp** indicates the editor, transcriber or encoder responsible for the decision not to provide any transcription of the text and hence the application of the <gap> tag.

<unclear> contains a word, phrase, or passage which cannot be transcribed with certainty because it is illegible or inaudible in the source. Attributes include:

- **reason** indicates why the material is hard to transcribe.
- **resp** indicates the individual responsible for the transcription of the letter, word or passage contained with the <unclear> element.

<del> contains a letter, word or passage deleted, marked as deleted, or otherwise indicated as superfluous or spurious in the copy text by an author, scribe, annotator or corrector. Attributes include:

- **type** classifies the type of deletion using any convenient typology.
- **status** may be used to indicate faulty deletions, e.g. strikeouts which include too much or too little text.
- **resp** signifies the editor or transcriber responsible for identifying the hand of the deletion.
- **hand** signifies the hand of the agent which made the deletion.
- **cert** signifies the degree of certainty ascribed to the identification of the hand of the deletion.

Encoders may choose to omit parts of the copy text for reasons ranging from illegibility of the source or impossibility of transcribing it, to editorial policy, e.g. a systematic exclusion of poetry or prose from an encoding. The full details of the policy decisions concerned should be documented in the TEI Header (see section 5.3 ('The Encoding Description') on p. 93). Each place in the text at which omission has taken place should be marked with a <gap> element, with optionally further information about the reason for the omission, its extent and the person or agency responsible for it, as in the following examples:

```
<gap desc="Prose commentary" reason="sampling"
      extent="120 lines" resp=PR>
```

```
... Their arrangement with respect to Jupiter and to each
other was as follows:
<gap desc="diagram" reason="sampling" extent="2 cm x 1 col">
That is, there were two starts on the easterly side and one
to the west; ...

<gap reason="illegible" desc="ink blot" extent="two words">

<gap reason='overwriting, illegible'
      resp='H1'
      extent='8 chars'>
```

The <add> and <del> elements may be used to record where words or phrases have been added or deleted in the copy text. They are not appropriate where longer passages have been added or deleted, which span several SGML elements; for these, the elements <addSpan> and <delSpan>, or other mechanisms described in section 18 ('Transcription of Primary Sources') on p. 443 must be used.

Additions to a text may be recorded for a number of reasons. Sometimes they are marked in a distinctive way in the source text, for example by brackets or insertion above the line (*supralinear* insertion), as in the following example, taken from a 19th century manuscript:

```
The story I am going to relate is true as to its main
facts, and as to the consequences <add
place='supralinear' resp='auth'>of these facts</add>
from which this tale takes its title.
```

The <add> element should not be used to mark editorial changes, such as supplying a word omitted by mistake from the source text or a passage present in another version. In these cases, either the <corr> or <supplied> tags should be used, as discussed above in section 6.5.1 ('Correction of Apparent Errors') on p. 141, and in section 18.1.3 ('Correction and Conjecture') on p. 447, respectively.

The <unclear> element is used to mark passages in the original which cannot be read with confidence, or about which the transcriber is uncertain for other reasons, as for example when transcribing a partially inaudible or illegible source. Its reason and resp attributes are used, as with the <gap> element, to indicate the cause of uncertainty and the person responsible for the conjectured reading.

For example:

```
<l>And where the sandy mountain Fenwick scald</>
<l><unclear cause="ink blot" resp=LB>The</unclear> sea between
   yet hence his pray'r prevail'd</>
```

or from a spoken text:

```
and then <unclear cause='passing truck'>marbled queen</unclear>
```

Where the material affected is entirely illegible or inaudible, the <gap> element discussed above should be used in preference.

The <del> element is used to mark material which is deleted in the source but which can still be read with some degree of confidence, as opposed to material which has been omitted by the encoder or transcriber either because it is entirely illegible or for some other reason. This is of particular importance in transcribing manuscript material, though deletion is also found in printed texts, sometimes for humorous purposes:

```
<l>One day I will sojourn to your shores</>
<l>I live in the middle of England</>
<l>But!</>
<l>Norway! My soul resides in your watery
   <del type='overstrike'>fiords fyords fiiords</del></>
<l>Inlets.</>
```

The type attribute may be used to distinguish different methods of deletion in manuscript or typescript material, as in this line from the typescript of Eliot's *Waste Land*:

```
<l><del type=overtyped>Mein</del> Frisch
   <del type=overstrike>schwebt</del> weht der Wind</>
```

Deletion in manuscript or typescript is often associated with addition:

```
<l><del type=overstrike>Inviolable</del>
   <add place=infralinear>Inexplicable</add>
   splendour of Corinthian white and gold</>
```

The <del> element should not be used where the deletion is such that material cannot be read with confidence, or read at all, or where the material has been omitted by the transcriber or editor for some other reason. Where the material cannot be read with confidence following deletion, the <unclear> tag should be used with the reason attribute indicated that the difficulty of transcription is due to deletion. Where material has been omitted by the transcriber or editor, this may be indicated by use of the <corr> (or <sic>) and <gap> elements. Observe that the distinction between recommended uses of the <del>, <corr> and <gap> tags parallels the distinction drawn between the <add>, <corr> and <supplied> tags in section 6.5.1 ('Correction of Apparent Errors') on p. 141 and section 18.1.3 ('Correction and Conjecture') on p. 447:

- where the correction is a deletion by a scribe or author in a manuscript or other primary source (typescript, proof or galley, etc.) then either <corr> (or <sic>) or <del> might be appropriate, depending on the circumstances. The <del> tag is more expressive, and may convey information on just how the deletion was performed (hand, place, etc.) which the <corr> tag cannot. See further the discussion in section 18.1.5 ('Substitutions') on p. 452.
- where the correction is a deletion by a transcriber or editor, correcting a perceived superfluity in the text but in circumstances where there is no clearly assertable reason for the superfluity (as a spurious addition) the <corr> tag should be used. The <del> tag should not be used in this case.
- where the correction is a deletion by a transcriber or editor, correcting a perceived superfluity in the text where there is a clearly assertable reason for the superfluity (as a spurious addition) the <gap> tag should be used with the reason attribute carrying the reason for the superfluity and hence the deletion of text. Neither the <del> nor <corr> tag should be used in these cases.

For any detailed transcription of a manuscript or typescript with more than trivial amounts of alteration, the reader should consult chapter 19 ('Critical Apparatus') on p. 467, and chapter 18 ('Transcription of Primary Sources') on p. 443.

These elements are formally defined as follows:

```
<!-- 6.5.3:  Other editorial tags                              -->
<!ELEMENT gap           - O  EMPTY                               >
<!ATTLIST gap                %a.global;
          desc               CDATA               #IMPLIED
          reason             CDATA               #IMPLIED
          resp               IDREF               %INHERITED
          hand               IDREF               %INHERITED
          agent              CDATA               #IMPLIED
          extent             CDATA               #IMPLIED      >
<!ELEMENT add           - -  (%specialPara;)                     >
<!ATTLIST add                %a.global;
          place              CDATA               #IMPLIED
          resp               IDREF               %INHERITED
          cert               CDATA               #IMPLIED
          hand               IDREF               %INHERITED    >
<!ELEMENT del           - -  (%phrase.seq;)                      >
<!ATTLIST del
          id                 ID                  #IMPLIED
          n                  CDATA               #IMPLIED
          lang               IDREF               %INHERITED
          rend               CDATA               #IMPLIED
          type               CDATA               #IMPLIED
```

```
            status              CDATA               'unremarkable'
            resp                IDREF               %INHERITED
            cert                CDATA               #IMPLIED
            hand                IDREF               %INHERITED     >
  <!ELEMENT unclear       - O   (%paraContent;)                    >
  <!ATTLIST unclear             %a.global;
            reason              CDATA               #IMPLIED
            resp                CDATA               %INHERITED
            cert                CDATA               #IMPLIED
            hand                IDREF               %INHERITED
            agent               CDATA               #IMPLIED       >
  <!-- This fragment is used in sec. 6.12                          -->
```

6.6 Simple Links and Cross References

Cross-references or links between one location in a document and another, or between one location and several others, may be encoded using the elements <ptr> and <ref>, as discussed in this section. These elements both "point" from one location in a document, the place that the element itself appears, to another (or to several), specified by the target attribute. Linkages of several other kinds are also provided for in these guidelines; see further chapter 14 ('Linking, Segmentation, and Alignment') on p. 331.

The pointing facility of these elements depends on the ability to supply a unique identifier for any element in the TEI scheme, using the global id attribute. Where the object or objects of a cross-reference are not identifiable in this way, either because they are located in a distinct SGML document or because no id attribute is available, the elements <xptr> or <xref> may be used instead.[5] Alternatively, if no explicit link is to be encoded, but it is simply required to mark the phrase as a cross-reference, the <ref> element may be used without a target attribute.

<ptr> defines a pointer to another location in the current document in terms of one or more identifiable elements. Attributes include:

- **target** specifies the destination of the pointer as one or more SGML identifiers

<ref> defines a reference to another location in the current document, in terms of one or more identifiable elements, possibly modified by additional text or comment. Attributes include:

- **target** specifies the destination of the reference as one or more SGML identifiers

The elements <ptr> and <ref> share, as members of the element class *pointer*, the following attributes:

type categorizes the pointer in some respect, using any convenient set of categories.
resp specifies the creator of the pointer.
crdate specifies when the pointer was created.
targType specifies the kinds of elements to which this pointer may point.
targOrder where more than one identifier is supplied as the value of the target attribute, this attribute specifies whether the order in which they are supplied is significant. Legal values are:

- ***Y*** Yes: the order in which IDREFs are specified as the value of a target attribute should be followed when combining the targeted elements.
- ***N*** No: the order in which IDREFs are specified as the value of a target attribute has no significance when combining the targeted elements.
- ***U*** Unspecified: the order in which IDREFs are specified as the value of a target attribute may or may not be significant.

evaluate specifies the intended meaning when the target of a pointer is itself a pointer. Sample values include:

- ***all*** if the element pointed to is itself a pointer, then the target of that pointer will be taken, and so on, until an element is found which is not a pointer.

[5] See chapter 14 ('Linking, Segmentation, and Alignment') on p. 331, for a discussion of these elements and the extended syntax they provide for "hypertext" links.

one if the element pointed to is itself a pointer, then its target (whether a pointer or not) is taken as the target of this pointer.

none no further evaluation of targets is carried out beyond that needed to find the element specified in the pointer's target.

The shared attributes of the two elements may be used in the same way; the difference between the elements is that while the <ptr> element is empty, the <ref> element may contain phrases specifying, or defining more exactly, the target of a cross reference, which form the content of the element. Since its content thus serves as a human-readable pointer, in the simplest case a <ref> element need not identify its target in any other way. For example:

```
See <ref>section 12 on page 34</ref>.
```

More usually, it will be desirable to identify the target of the cross-reference using the target attribute, so that processing software can access it directly, for example to implement a linkage or to generate an appropriate reference. Assuming that section 12 in the previous example has been tagged <div1 id=SEC12>, the same cross reference might more exactly be encoded as

```
See especially <ref target=SEC12>section 12 on page 34</ref>.
```

If the text for the cross reference is to be generated according to a fixed pattern, or if no text is to appear in the body of the cross reference, the <ptr> element would be used as follows:

```
See in particular <ptr target=sec12>.
```

A cross-reference may point to any number of locations simultaneously, simply by giving more than one identifier as the value of its target attribute. This may be particularly useful where an analytic index is to be encoded, as in the following example:

```
<list>
<item>Saints aid rejected in mel. <ptr target=p299>
<item>Sallets censured <ptr target="p143 p144">
<item>Sanguine mel. signs <ptr target="p263 p312 p332">
<item>Scilla or sea onyon, a purger of mel. <ptr target=p442>
...
```

Here the targets of the cross references are simply page numbers; it is assumed that corresponding elements with identifiers p299 p143, etc. have been provided in the body of the text. If it is desired to check that the target elements are of a particular type, the targType (target type) attribute may be specified:

```
<list>
<item>Saints aid rejected in mel <ptr target=p299 targType=pb>
<item>Sallets censured <ptr target="p143 p144" targType=pb>
...
```

Here, a processing application can check that the elements with identifiers p299, p143, and p144 are all <pb> (page-break) elements. It is a semantic error in a text if the targets given do not match the values specified on a targType attribute.

The type and resp attributes may be used, as elsewhere, to categorize the cross reference according to any system of importance to the encoder and to supply a code identifying the person or agency responsible for the cross reference. If bibliographic references require special processing (e.g. in order to provide a consistent short-form reference), they might be tagged thus:

```
Similar forms, often called <term rend='ldquo
rdquo'>rewriting systems</term>, have a long
history among mathematicians, but the specific form
of <ptr targType=fig target=Fig22> was first
studied extensively by Chomsky <ptr type=bibliog
targType='bibl bibl.struct bibl.full' target=Chom59>.
```

Here type=bibliog signals for the processing appropriate to a bibliographic reference, while "targType='bibl bibl.struct bibl.full"' restricts the legal targets to bibliographic elements, and target=Chom59 indicates which bibliographic element actually is being referred to. For further discussion of bibliographic references, see section 6.10.3 ('Bibliographic Pointers ') on p. 175.

If the order in which the objects of a multi-headed cross reference are specified is of importance, the targOrder (target order) attribute should be specified.

```
<p>The following discussions of this topic should
be consulted for further information:
<ptr target='ch3 sec332 sec45 sec722' targOrder=Y>
```

The <ptr> and <ref> tags have many applications in addition to the simple cross-referencing facilities illustrated in this section. In conjunction with the analytic tools discussed in chapters 14 ('Linking, Segmentation, and Alignment') on p. 331, 15 ('Simple Analytic Mechanisms') on p. 381, and 16 ('Feature Structures') on p. 397, they may be used to link analyses of a text to their object, to combine corresponding segments of a text, or to align segments of a text with a temporal or other axis or with each other.

These elements are formally defined as follows:

```
<!-- 6.6:  Simple cross references                          -->
<!ENTITY % a.pointer '
          type              CDATA              #IMPLIED
          resp              CDATA              #IMPLIED
          crdate            CDATA              #IMPLIED
          targType          NAMES              #IMPLIED
          targOrder         (Y | N | U)        U
          evaluate          (all | one | none) #IMPLIED'     >
<!ELEMENT ptr           - O  EMPTY                            >
<!ATTLIST ptr                %a.global;
                             %a.pointer;
          target            IDREFS             #REQUIRED      >
<!ELEMENT ref           - -  (%paraContent)                   >
<!ATTLIST ref                %a.global;
                             %a.pointer;
          target            IDREFS             #IMPLIED       >
<!-- This fragment is used in sec. 6.12                      -->
```

6.7 Lists

The following elements are provided for the encoding of lists, their constituent items, and the labels or headings associated with them:

<list> contains any sequence of items organized as a list. Attributes include:

type describes the form of the list. Suggested values include:

ordered list items are numbered or lettered.

bulleted list items are marked with a "bullet" or other typographic device.

simple list items are not numbered or bulleted.

gloss each list item glosses some term or concept, which is given by a <label> element preceding the list item.

<item> contains one component of a list. Attributes include:

n indicates the number (or letter, or other enumerator) borne by this list item in the copy text.

<label> contains the label associated with an item in a list; in glossaries, marks the term being defined.

<head> contains any heading, for example, the title of a section, or the heading of a list or glossary.

<headLabel> contains the heading for the label or term column in a glossary list or similar structured list.

<headItem> contains the heading for the item or gloss column in a glossary list or similar structured list.

The <list> element should be used to mark any kind of *list*: numbered, lettered, bulleted, or unmarked. Lists formatted as such in the copy text should in general be encoded using this

element, with an appropriate value for the type attribute. Lists given as run-on text may also be encoded using this element, where this is felt to be appropriate.

Each distinct item in the list should be encoded as a distinct <item> element. If the numbering or other identification for the items in a list is unremarkable and may be reconstructed by any processing program, no enumerator need be specified. If however an enumerator is retained in the encoded text, it may be supplied either by using the n attribute on the <item> element, or by using a <label> element. The following examples are thus equivalent:

```
I will add two facts, which have seldom occurred in
the composition of six, or at least of five quartos.
<list type=ordered rend=runon>
<label>(1)</><item>My first rough manuscript, without any
intermediate copy, has been sent to the press.</item>
<label>(2)</><item>Not a sheet has been seen by any human
eyes, excepting those of the author and the printer:
the faults and the merits are exclusively my own.</item>
</list>
```

```
I will add two facts, which have seldom occurred in
the composition of six, or at least of five quartos.
<list type=ordered rend=runon>
<item n='1'>My first rough manuscript, without any
intermediate copy, has been sent to the press.</>
<item n='2'>Not a sheet has been seen by any human
eyes, excepting those of the author and the printer:
the faults and the merits are exclusively my own.</>
</list>
```

The two styles may not be mixed in the same list: if one item is preceded by a label, all must be.

A list need not necessarily be displayed in list format. For example,

```
On those remote pages it is written that animals are
divided into <list><item n='a'>those that belong to the
Emperor,</><item n='b'> embalmed ones, </><item n='c'> those
that are trained, </><item n='d'> suckling pigs, </><item n='e'>
mermaids,  </><item n='f'> fabulous ones, </><item n='g'> stray
dogs, </><item n='h'> those that are included in this
classification, </><item n='i'> those that tremble as if they
were mad, </><item n='j'> innumerable ones, </><item n='k'> those
drawn with a very fine camel's-hair brush, </><item n='l'>
others, </><item n='m'> those that have just broken a flower
vase, </><item n='n'> those that resemble flies from a
distance.</></list>
```

A list may be given a heading or title, for which the <head> element should be used, as in the next example, which also demonstrates simple use of the <label> element to mark a tabular or glossary list in which each item is associated with a word or phrase rather than a numeric or alphabetic enumerator:

```
<list type=gloss>
<head>Report of the conduct and progress of Ernest Pontifex.
Upper Vth form &mdash half term ending Midsummer 1851</head>
<label>Classics</><item>Idle listless and unimproving</>
<label>Mathematics</><item>ditto</>
<label>Divinity</><item>ditto</>
<label>Conduct in house</><item>Orderly</>
<label>General conduct</><item>Not satisfactory, on account
of his great unpunctuality and inattention to duties</>
</list>
```

In such a list, the individual items have internal structure. In complex cases, where list items contain many components, the list is better treated as a *table*, on which see chapter 22 ('Tables, Formulae, and Graphics') on p. 523. A particularly important instance of the simple two-column

table is the "glossary list", which should be marked by the tag <list type=gloss>. In such lists, each <label> element contains a term and each <item> its gloss; it is a semantic error for a list tagged with type=gloss not to have labels. For example:

```
<list type=gloss>
<head>Unit Three -- Vocabulary</head>
<label lang=la>acerbus, -a, -m       </><item>bitter, harsh</>
<label lang=la>ager, agr&imacr;, M.  </><item>field</>
<label lang=la>audi&omacr;, &imacr;re,
       &imacr;v&imacr;, &imacr;tus   </><item>hear, listen (to)</>
<label lang=la>bellum, -&imacr;, N.  </><item>war</>
<label lang=la>bonus, -a, -um        </><item>good</>
<!-- etc. -->
</list>
```

Additionally, the <term> and <gloss> elements discussed in section 6.3.4 ('Terms, Glosses, and Cited Words') on p. 130 might be used to make explicit the role that each column in the glossary list has, as follows:

```
<list type=gloss>
<head>Unit Three -- Vocabulary</head>
<label><term lang=la>acerbus, -a, -m</term></>
<item><gloss>bitter, harsh</gloss></item>
<label><term lang=la>ager, agr&imacr;, M. </term></>
<item><gloss>field</gloss></item>
<label><term lang=la>audi&omacr;, &imacr;re,
       &imacr;v&imacr;, &imacr;tus  </term></>
<item><gloss>hear, listen (to)</gloss></item>
<label><term lang=la>bellum, -&imacr;, N. </term></>
<item><gloss>war</gloss></item>
<label><term lang=la>bonus, -a, -um</term></>
<item><gloss>good</gloss></item>
<!-- etc. -->
</list>
```

Note in the above examples the use of the global lang attribute to specify on the <label> (or <term>) element what language the term is from. For further discussion of the lang attribute see section 3.5 ('Global Attributes') on p. 42, and section 4.2 ('Shifting Among Character Sets') on p. 74. A more elaborate markup for this glossary would distinguish the headword forms from the grammatical information (principal parts and gender), using tags described more fully in chapters 13 ('Terminological Databases') on p. 311 or 12 ('Print Dictionaries') on p. 269.

In addition to the <head> element used to supply a title or heading for the whole list, headings for the two columns of a glossary-style list may be specified using the two special elements <headLabel> and <headItem>:

```
The simple, straightforward statement of an idea is
preferable to the use of a worn-out expression.
<list type=gloss>
<headLabel>TRITE</>
<headItem>SIMPLE, STRAIGHTFORWARD</>
<label>bury the hatchet  </><item>stop fighting, make peace</>
<label>at loose ends     </><item>disorganized</>
<label>on speaking terms </><item>friendly</>
<label>fair and square   </><item>completely honest</>
<label>at death's door   </><item>near death</>
</list>
```

The elements <label>, <head>, <headLabel>, and <headItem> may contain only phrase-level elements. The <item> element however may contain paragraphs or other "chunks", including other lists. In this example, a glossary list contains two items, each of which is itself a simple list:

```
<list type=gloss><label>EVIL</label>
<item><list type=simple>
```

```
   <item>I am cast upon a horrible desolate island, void
         of all hope of recovery.</item>
   <item>I am singled out and separated as it were from
        all the world to be miserable.</item>
   <item>I am divided from mankind &mdash a solitaire; one
          banished from human society.</item>
     </list> <!-- end of first nested list --></item>
<label>GOOD</label>
<item><list type=simple>
     <item>But I am alive; and not drowned, as all my
           ship's company were.</item>
     <item>But I am singled out, too, from all the ship's
          crew, to be spared from death...</item>
     <item>But I am not starved, and perishing on a barren place,
          affording no sustenances....</item>
     </list><!-- end of second nested list --></item>
</list><!-- end of glossary list -->
```

Lists of different types may be nested to arbitrary depths in this way.
The formal declarations for lists and list items are as follows.

```
<!-- 6.7:  Lists and List Items                                  -->
<!ELEMENT list          - -  (head?, ( (item+) | (headLabel?,
                             headItem?, (label, item)+)))       >
<!ATTLIST list               %a.global;
          type               CDATA              simple          >
<!ELEMENT item          - O  (%specialPara;)                    >
<!ATTLIST item
          id                 ID                 #IMPLIED
          lang               IDREF              %INHERITED
          rend               CDATA              #IMPLIED
          n                  CDATA              #IMPLIED        >
<!ELEMENT label         - o  (%phrase.seq;)                     >
<!ATTLIST label              %a.global;                         >
<!ELEMENT head          - O  (%paraContent;)                    >
<!ATTLIST head               %a.global;
          type               CDATA              #IMPLIED        >
<!ELEMENT headLabel     - O  (%phrase.seq;)                     >
<!ATTLIST headLabel          %a.global;                         >
<!ELEMENT headItem      - O  (%phrase.seq;)                     >
<!ATTLIST headItem           %a.global;                         >
<!-- This fragment is used in sec. 6.12                        -->
```

6.8 Notes, Annotation, and Indexing

6.8.1 Notes and Simple Annotation

The following elements are provided for the encoding of discursive notes, either already present in the copy text or supplied by the encoder:

<note> contains a note or annotation. Attributes include:

- **n** indicates the symbol or number used to mark the note's point of attachment to the main text.
- **type** describes the type of note.
- **resp** indicates who is responsible for the annotation: author, editor, translator, etc. Sample values include:
  - ***auth[or]*** note originated with the author of the text.
  - ***ed[itor]*** note added by the editor of the text.
  - ***comp[iler]*** note added by the compiler of a collection.

tr[anslator] note added by the translator of a text.

transcr[iber] note added by the transcriber of a text into electronic form.

(initials) note added by the individual indicated by the initials.

place indicates where the note appears in the source text. Sample values include:

foot note appears at foot of page.

end note appears at end of chapter or volume.

inline note appears as a marked paragraph in the body of the text.

left note appears in left margin.

right note appears in right margin.

interlinear note appears between lines of the text.

app[aratus] note appears in the apparatus at the foot of the page.

anchored indicates whether the copy text shows the exact place of reference for the note. Legal values are:

yes copy text indicates the place of attachment for the note.

no copy text indicates no place of attachment for the note.

target indicates the point of attachment of a note, or the beginning of the span to which the note is attached.

targetEnd points to the end of the span to which the note is attached, if the note is not embedded in the text at that point.

A *note* is any additional comment found in a text, marked in some way as being out of the main textual stream. All notes should be marked using the same tag, <note>, whether they appear as block notes in the main text area, at the foot of the page, at the end of the chapter or volume, in the margin, or in some other place.

Notes may be in a different hand or typeface, may be authorial or editorial, and may have been added later. Attributes may be used to specify these and other characteristics of notes, as detailed below.

Where possible, the body of a note should be inserted in the text at the point at which its identifier or mark first appears. This may not be possible for example with marginal notes, which may not be anchored to an exact location. For simplicity, it may be adequate to position marginal notes before the relevant paragraph or other element. In some cases, however, it may be desirable to transcribe notes not at their point of attachment to the text but at their point of appearance (at the end of the volume, or the end of the chapter — not, in general, when the notes appear at the foot of the page); in this case the target and targetEnd attributes should be used to specify the point of attachment. In some cases, the note is explicitly attached not to a point but to a span of text; for a full discussion of pointing to points and spans in the text, see section 6.6 ('Simple Links and Cross References') on p. 147.

Examples:

```
<l>The self-same moment I could pray
<l>And from my neck so free
<l>The albatross fell off, and sank
<l>Like lead into the sea.
<note type=auth place=margin>The spell begins to break</note>
```

```
Collections are ensembles of distinct entities or objects
of any sort.<note place=foot n=1>We explain below why we use
the uncommon term <mentioned>collection</mentioned>
instead of the expected <mentioned>set</mentioned>.
Our usage corresponds to the <mentioned>aggregate</mentioned> of many
mathematical writings and to the sense of <mentioned>class</mentioned>
found in older logical writings.
</note>
The elements ...
```

In addition to transcribing notes from the copy text, researchers may wish to annotate the electronic text itself, by attaching analytic notes in some structured vocabulary to particular passages of text, e.g. to specify the topics or themes of a text. The empty element is

provided for such applications; it is available only when the additional tag set for simple analysis is selected (see section 15.3 ('Spans and Interpretations') on p. 387).

The formal declarations for the <note> element is this:

```
<!-- 6.8.1:  Annotation                                          -->
<!ELEMENT note          - O  (%specialPara;)                      >
<!ATTLIST note
          id                ID                  #IMPLIED
          lang              IDREF               %INHERITED
          rend              CDATA               #IMPLIED
          n                 CDATA               #IMPLIED
          type              CDATA               #IMPLIED
          resp              CDATA               #IMPLIED
          place             CDATA               'unspecified'
          anchored          (yes | no)          yes
          target            IDREFS              #IMPLIED
          targetEnd         IDREFS              #IMPLIED       >
<!-- ... declarations from section 6.8.2                        -->
<!--     (Index Entries)                                        -->
<!--     go here ...                                            -->
<!-- This fragment is used in sec. 6.12                         -->
```

6.8.2 Index Entries

Machine-readable versions of existing texts rarely reproduce any index published with the copy text. Should a printed index be transcribed, the <div1> tag or a tag at an appropriate level should be used to demarcate the index, and the index itself may be transcribed as a structured list or table.

It is convenient, however, to be able to generate a new index from a machine-readable text, whether the text is being written for the first time with the tags here defined or was transcribed from some other source. The <index> tag is provided for this purpose; it may be useful for marking points of particular interest for whatever reason, and not merely for generating printed indexes for a printed version of the text. The <divGen> element indicates the point at which an index, or any other generated text (e.g. a table of contents), is to appear in the output of a text production process.

<index> marks a location to be indexed for whatever purpose. Attributes include:

- **level1** gives the form under which the index entry is to be made.
- **level2** gives the second-level form, if any.
- **level3** gives the third-level form, if any.
- **level4** gives the fourth-level form, if any.
- **index** indicates which index (of several) the index entry belongs to.

<divGen> indicates the location at which a textual division generated automatically by a text-processing application is to appear. Attributes include:

- **type** specifies what type of generated text division (e.g. index, table of contents, etc.) is to appear. Sample values include:
  - ***index*** an index is to be generated and inserted at this point.
  - ***toc*** a table of contents
  - ***figlist*** a list of figures
  - ***tablist*** a list of tables

The tag <index> associates up to four levels of index terms with a specific point in the text. The index terms are supplied in attributes named level1, level2, level3, and level4. An index attribute associates the entry with a particular index, so multiple indices are possible.

All index terms must be supplied as attribute values; no part of the text itself is taken as a term. This may require words or phrases to be repeated, as illustrated below; it also allows spelling to be normalized, as the example shows:

```
The students understand procedures for Arabic lemmatisation
```

```
<index level1='Arabic lemmatization'>and are beginning
to build parsers.
```

The <divGen> element marks the place at which an index generated from the <index> elements should be inserted into the output of a processing program; typically, this will be at some point within the back matter of the document; its type attribute should be used to specify which index is to be generated, and its n attribute to specify a name for the index:

```
<back>
<div><head>Examples</head>
<p> ...
</div>
<div><head>Bibliography</head>
<list.bibl> ... </list.bibl>
</div>
<divGen type='index 1' n='Index Nominum'>
<divGen type='index 2' n='Index Rerum'  >
</back>
```

The formal declaration for these elements is as follows.

```
<!-- 6.8.2:  Index Entries                                    -->
<!ELEMENT index         - O  EMPTY                             >
<!ATTLIST index              %a.global;
          index              CDATA              #IMPLIED
          level1             CDATA              #REQUIRED
          level2             CDATA              #IMPLIED
          level3             CDATA              #IMPLIED
          level4             CDATA              #IMPLIED       >
<!ELEMENT divGen        - O  EMPTY                             >
<!ATTLIST divGen             %a.global;
          type               CDATA              #IMPLIED       >
<!-- This fragment is used in sec. 6.8.1                      -->
```

6.9 Reference Systems

By 'reference system' we mean the system by which names or references are associated with particular passages of a text (e.g. 'Ps. 23:3' for the third verse of Psalm 23 or 'Amores 2.10.7' for Ovid's *Amores*, book 2, poem 10, line 7). Such names make it possible to mark a place within a text and enable other readers to find it again. A reference system may be based on structural units (chapters, paragraphs, sentences; stanza and verse), typographic units (page and line numbers), or divisions created specifically for reference purposes (chapter and verse in Biblical texts). Where one exists, the traditional reference system for a text should be preserved in an electronic transcript of it, if only to make it easier to compare electronic and non-electronic versions of the text.

Reference systems may be recorded in TEI-encoded texts in any of the following ways:

- where a reference system exists, and is based on the same logical structure as that of the text's SGML markup, the reference for a passage may be recorded as the value of the global id or n attribute on an appropriate tag, or may be constructed by combining attribute values from several levels of tags, as described below in section 6.9.1 ('Using the ID and N Attributes') on p. 156.
- where there is no pre-existing reference system, the global id or n attributes may be used to construct one (e.g. collections and corpora created in electronic form), as described below in section 6.9.2 ('Creating New Reference Systems') on p. 157.
- where a reference system exists which is not based on the same logical structure as that of the text's SGML markup, (for example, one based on the page and line numbers of particular editions of the text rather than on the structural divisions of it), then a distinct logical structure representing the reference system may be encoded as a concurrent markup stream, as described in section 31.6 ('Concurrent Markup for Pages and Lines') on p. 637.

- where a reference system exists which does not correspond to any particular logical structure, or where the logical structure concerned is of no interest to the encoder except as a means of supporting the referencing system, then references may be encoded by means of <milestone> elements, which simply mark points in the text at which values in the reference system change, as described below in section 6.9.3 ('Milestone Tags') on p. 158.

The specific method used to record traditional or new reference systems for a text should be declared in the TEI header, as further described in section 6.9.4 ('Declaring Reference Systems') on p. 161 and in chapter 32 ('Algorithm for Recognizing Canonical References') on p. 641.

When a text has no pre-existing associated reference system of any kind, these Guidelines recommend as a minimum that at least the page boundaries of the source text be marked using one of the methods outlined in this section. Retaining page breaks in the markup is also recommended for texts which have a detailed reference system of their own. Line breaks in prose texts may be, but need not be, tagged. [6]

6.9.1 Using the ID and N Attributes

When traditional reference schemes represent a hierarchical structuring of the text which mirrors that of the SGML document, the n attribute defined for all elements may be used to indicate the traditional identifier of the relevant structural units. The n attribute may also be used to record the numbering of sections or list items in the copy text if the copy-text numbering is important for some reason, for example because the numbers are out of sequence.

For example, a traditional reference to Ovid's *Amores* might be 'Amores 2.10.7'—book 2, poem 10, line 7. Book, poem, and line are structural units of the work and will therefore be tagged in any case. (See chapter 9 ('Base Tag Set for Verse ') on p. 213 for a discussion of structural units in verse collections.) In such cases, it is convenient to record traditional reference numbers of the structural units using the n attribute. The relevant tags for our example would be:

```
<div0 n=Amores> ...
    <div1 n='2' type=book> ...
         <div2 n='10' type=poem> ...
              <l n='7'> ...
```

One may also place the entire standard reference for each portion of the text into the appropriate value for the n attribute, though for obvious reasons this takes more space in the file:

```
<div0 n=Amores> ...
    <div1 n='Amores 2' type=book> ...
         <div2 n='Amores 2.10' type=poem> ...
              <l n='Amores 2.10.7'> ...
```

If the names used by the traditional reference system can be formulated as SGML identifiers, then the references can be given as values for the id attribute; this requires that the reference be given without internal spaces, begin with a letter, and contain no characters other than letters, digits, hyphens, and full stops.[7] Unlike values for the n attribute, values for the id attribute must be unique throughout the document. Our example then looks like this:

```
<div0 id=Amores> ...
    <div1 id='Am.2' type=book> ...
         <div2 id='Am.2.10' type=poem> ...
              <l id='Am.2.10.7'> ...
```

To document the usage and to allow automatic processing of these standard references, it is recommended that the TEI header be used to declare whether standard references are recorded in the n or id attributes and which elements may carry standard references or portions of them. For examples of declarations for the reference systems just shown, see section 6.9.4 ('Declaring Reference Systems') on p. 161.

[6] Many encoders find it convenient to retain the line breaks of the original during data entry, to simplify proof-reading, but this may be done without inserting a tag for each line break of the original.

[7] The legal form of identifiers depends in part on the SGML declaration. With appropriate modifications in the declaration, other characters may be made legal in identifiers; this is allowed though not encouraged in TEI-conformant documents.

Using the n attribute one can specify only a single standard referencing system, a limitation not without problems, since some editions may define structural units differently and thus create alternative reference systems. For example, another edition of the *Amores* considers poem 10 a continuation of poem 9, and therefore would specify the same line as 'Amores 2.9.31'. In order to record both of these reference systems, one must either use the id or n attribute for one, with competing systems in concurrent markup hierarchies, as discussed below in section 31.6 ('Concurrent Markup for Pages and Lines') on p. 637, or else use the <milestone> tags described in section 6.9.3 ('Milestone Tags') on p. 158.

6.9.2 Creating New Reference Systems

If a text has no canonical reference system of its own, a reference system, if needed, may be derived from the structure of the electronic text, specifically from the SGML markup of the text. As with any reference system intended for long-term use, it is important to see the reference as an established, unchanging point in the text. Should the text be revised or rearranged, the reference-system identifiers associated with any bit of text must stay with that bit of text, even if it means the reference numbers fall out of sequence. (A new reference system may always be created beside the old one if out-of-sequence numbers must be avoided.)

The global attributes n and id may be used to assign reference identifiers to segments of the text. Identifiers specified by either attribute apply to the entire element for which they are given. SGML enforces uniqueness on ID attributes within a single document, and ID values must begin with a letter. No such restrictions are made on the values of n attributes.

A convenient method of mechanically generating unique values for id or n attributes based on the SGML structure of the document is to construct, for each element, a *domain-style address* comprising a series of components separated by full stops, with one component for each level of the SGML document hierarchy. Two methods may be use. In the *typed path* form of identifier, each component in the identifier takes the form *element-type* '-' *number*. The element name specifies what type of element to be sought, and the number specifies which occurrence of that element type is to be selected. (The hyphen and number may be omitted if there is only one element of the given type.) In the *untyped path* form of identifier, each component consists of a number, indicating which element in the sequence of nodes at each level is to be selected. A fixed prefix beginning with a letter may be used to make the untyped path legal as an SGML ID value.

Identifiers generated with these methods should use the <text> segment as their starting point, rather than the <tei.2> or <body> elements, since the <tei.2> element need be taken as a starting point only if identifiers need to be generated for the <teiHeader>, which is not usually the case, and using the <body> element as a root would prevent assignment of identifiers for the front and back matter. The component corresponding to the root element can be omitted from identifiers, if no confusion will result. In collections and corpora, the component corresponding to the root may be replaced by the unique identifier assigned to the text or sample.

In the following example, each element within the <text> element has been given a typed-path identifier as its id value, and an untyped-path identifier as its n value; the latter are prefixed with the string 'AB', which may be imagined to be the general identifier for this text.

```
<tei.2>
  <teiHeader> ... </teiHeader>
  <text id='text-1' n='AB'>
    <front id='front' n='AB.1'>
      <div       id='front.div-1'     n='AB.1.1'> ... </div>
      <titlePage id='front.titlePage' n='AB.1.2'> ... </>
      <div       id='front.div-2'     n='AB.1.3'> ... </div>
    </front>
    <body id='body' n='AB.2'>
      <p id='body.p-1' n='AB.2.1'> ... </p>
      <p id='body.p-1' n='AB.2.2'> ... </p>
      <div id='body.div-1' n='AB.2.3'>
        <head id='body.div-1.head' n='AB.2.3.1'> ... </head>
        <p    id='body.div-1.p-1'  n='AB.2.3.2'> ... </p>
        <p    id='body.div-1.p-2'  n='AB.2.3.3'> ... </p>
```

```
    </div>
  <div id='body.div-2' n='AB.2.4'>
    <head id='body.div-2.head' n='AB.2.4.1'> ... </head>
    <p    id='body.div-2.p-1'  n='AB.2.4.2'> ... </p>
    <p    id='body.div-2.p-2'  n='AB.2.4.3'> ... </p>
    </div>
</body>
```

The typed and untyped path methods are convenient, but are in no way required for anyone creating a reference system.

If the **id** attribute is used to record the reference identifiers generated, each value should record the entire path. If the **n** attribute is used, each value may record either the entire path or only the subpath from the SGML parent element. The attribute used, the elements which can bear standard reference identifiers, and the method for constructing standard reference identifiers, should all be declared in the header as described in section 5.3.5 (‘The Reference System Declaration ’) on p. 100.

When the hierarchy of the SGML-encoded document and that of the reference system differ (e.g. for reference systems based on page and line numbers) or when more than one reference system is to be encoded, the encoder may choose to represent the alternative reference system(s) as elements in one or more concurrent document hierarchies. For an introduction to the concept of concurrent hierarchies, see the discussion of the **CONCUR** feature in section 2.5.2 (‘Concurrent Structures ’) on p. 24. For further discussion of this and other mechanisms, see chapter 31 (‘Multiple Hierarchies’) on p. 633.

6.9.3 Milestone Tags

If concurrent markup is not desired (e.g. because the available SGML parser does not support the CONCUR feature), or if the desired reference system does not correspond to any particular structural hierarchy, it may be more convenient to mark up changes in the reference system by using one or more of the following *milestone* elements:

<milestone> marks the boundary between sections of a text, as indicated by changes in a standard reference system. Attributes include:

- **ed** indicates which edition or version the milestone applies to.
- **n** indicates the new number or other value for the unit which changes at this milestone.
- **unit** indicates what kind of section is changing at this milestone. Suggested values include:
  - ***page*** page breaks in the reference edition.
  - ***column*** column breaks.
  - ***line*** line breaks.
  - ***book*** any units termed ‘book’, ‘liber’, etc.
  - ***poem*** individual poems in a collection.
  - ***canto*** cantos or other major sections of a poem.
  - ***stanza*** stanzas within a poem, book, or canto.
  - ***act*** acts within a play.
  - ***scene*** scenes within a play or act.
  - ***section*** sections of any kind.
  - ***absent*** passages not present in the reference edition.

<pb> marks the boundary between one page of a text and the next in a standard reference system. Attributes include:

- **ed** indicates the edition or version in which the page break is located at this point
- **n** indicates the number or other value associated with the page which follows the point of insertion of this **<pb>**.

<lb> marks the start of a new (typographic) line in some edition or version of a text. Attributes include:

- **ed** indicates the edition or version in which the line break is located at this point
- **n** indicates the number or other value associated with the line which follows the point of insertion of this **<lb>**.

<cb> marks the boundary between one column of a text and the next in a standard reference system. Attributes include:

ed indicates the edition or version in which the column break is located at this point

n indicates the number or other value associated with the column which follows the point of insertion of this <cb> element.

These elements simply mark the points in a text at which some category in a reference system changes. They have no content but subdivide the text into regions, rather in the same way as milestones divide a road into segments. The elements <pb>, <cb>, and <lb> are provided to mark specific types of milestone, namely page, column, and line boundaries, as further described in chapter 18 ('Transcription of Primary Sources') on p. 443. No SGML validation of a reference system based on <milestone> tags is possible, so it will be the responsibility of the encoder or the application software to ensure that milestone tags occur in a correct order.

Milestone tags may be useful where a text has two competing structures. For example, many English novels were first published as serial works, individual parts of which do not always contain a whole number of chapters. An encoder may decide to represent the chapter-based structure using <div1> elements, with <milestone> elements to mark the points at which individual parts end; or the reverse. Thus, an encoding in which chapters are regarded as more important than parts might encode some work in which chapter three begins in part one and is concluded in part two as follows:

```
<milestone unit=part>
<div1 n=1 type=chapter>
<!-- text of chapter 1 here -->
</div1>
<div1 n=2 type=chapter>
<!-- text of chapter 2 here -->
</div1>
<div1 n=3>
<!-- part of text of chapter 3 here -->
<milestone unit=part>
<!-- remainder of text of chapter 3 here -->.
</div1>
<!-- ... -->
```

An encoding of the same work in which parts are regarded as more important than chapters might begin as follows:

```
<div1 n=1 type=part>
<milestone unit=chapter>
<!-- text of chapter 1 here -->
<milestone unit=chapter>
<!-- text of chapter 2 here -->
<milestone unit=chapter>
<!-- part of text of chapter 3 here -->
</div1><div1 n=2 type=part>
<!-- remainder of text of chapter 3 here -->.
<milestone unit=chapter>
<!-- ... -->
```

Milestone tags also make it possible to record the reference systems used in a number of different editions of the same work. The reference system of any one edition can be recreated from a text in which all are marked by simply ignoring all elements that do not specify that edition on their ed attribute.

As a simple example, assuming that edition E1 of some collection of poems regards the first two poems as constituting the first book, while edition E2 regards the first poem as prefatory, a markup scheme like the following might be adopted:

```
<milestone ed=E1 unit=work>
<milestone ed=E2 unit=work>
```

```
<milestone ed=E1 unit=book>
<milestone ed=E1 unit=poem>
<milestone ed=E2 unit=poem>
 <!-- text of first poem here -->
<milestone ed=E2 unit=book>
<milestone ed=E1 unit=poem>
<milestone ed=E2 unit=poem>
 <!-- text of second poem here -->
```

In this case no `n` value is specified, since the numbers rise predictably and the application can keep a count from the start of the document, if desired.

The value of the `n` attribute may but need not include the identifiers used for any larger sections. That is, either of the following styles is legitimate:

```
<milestone ed=E1 unit=work n='Amores'>
<milestone ed=E1 unit=book n=1>
<milestone ed=E1 unit=poem n=1>
... <!-- text of Amores 1.1 -->
<milestone ed=E1 unit=poem n=2>
... <!-- text of Amores 1.2 -->
<milestone ed=E1 unit=book n=3>
...
```

or

```
<milestone ed=E1 unit=work n='Amores'>
<milestone ed=E1 unit=book n=1>
<milestone ed=E1 unit=poem n='1.1'>
... <!-- text of Amores 1.1 -->
<milestone ed=E1 unit=poem n='1.2'>
... <!-- text of Amores 1.2 -->
<milestone ed=E1 unit=book n='1.3'>
...
```

When using `<milestone>` tags, line numbers may be supplied for every line or only periodically (every fifth, every tenth line). The latter may be simpler; the former is more reliable.

The style of numbering used in the values of `n` is unrestricted: for the example above, `I.i`, `I.ii`, and `I.iii` could have been used equally well if preferred. The special value `unnumbered` should be reserved for marking sections of text which fall outside the normal numbering system (e.g. chapter heads, poem numbers, titles, or speaker attributions in a verse drama).

Because the `ed` attribute is unrestricted, no change need be made to the document type declaration of a file before adding tags to describe a new reference system. (The value of `ed` may be restricted to a defined set of edition symbols by using the techniques described in chapter 29 ('Modifying the TEI DTD') on p. 619.)

See below, section 6.9.4 ('Declaring Reference Systems') on p. 161, for examples of declarations for the reference systems just shown.

The milestone elements are formally defined as follows:

```
<!-- 6.9.3:  Milestone tags                                    -->
<!ELEMENT milestone     - O  EMPTY                               >
<!ATTLIST milestone
          id                ID                  #IMPLIED
          lang              IDREF               %INHERITED
          rend              CDATA               #IMPLIED
          ed                CDATA               #IMPLIED
          n                 CDATA               #IMPLIED
          unit              CDATA               #REQUIRED      >
<!ELEMENT pb            - O  EMPTY                               >
<!ATTLIST pb
          id                ID                  #IMPLIED
          lang              IDREF               %INHERITED
```

```
          rend              CDATA              #IMPLIED
          ed                CDATA              #IMPLIED
          n                 CDATA              #IMPLIED       >
<!ELEMENT lb           - O  EMPTY                             >
<!ATTLIST lb
          id                ID                 #IMPLIED
          lang              IDREF              %INHERITED
          rend              CDATA              #IMPLIED
          ed                CDATA              #IMPLIED
          n                 CDATA              #IMPLIED       >
<!ELEMENT cb           - O  EMPTY                             >
<!ATTLIST cb
          id                ID                 #IMPLIED
          lang              IDREF              %INHERITED
          rend              CDATA              #IMPLIED
          ed                CDATA              #IMPLIED
          n                 CDATA              #IMPLIED       >
<!-- This fragment is used in sec. 6.12                      -->
```

6.9.4 Declaring Reference Systems

Whatever kind of reference system is used in an electronic text, it is recommended that the TEI header contain a description of its construction in the <refsDecl> element described in section 5.3.5 ('The Reference System Declaration ') on p. 100. As described there, the declaration may consist either of a formal declaration using the <step> tag or an informal description in prose. The former is recommended because unlike prose it can be processed by software.

The three examples given in section 6.9.1 ('Using the ID and N Attributes') on p. 156 would be declared as follows. The first example encodes the standard references for Ovid's *Amores* one level at a time, using the n attribute on the <div0>, <div1>, <div2>, and <l> tags. The header for such an encoding should look something like this:

```
<teiHeader>
    <fileDesc> ...
    </fileDesc>
    <encodingDesc>
         ...
         <refsDecl>
              <step refunit='work' delim=' '
                    from='DESCENDANT (1 DIV0 N %1)' to='DITTO' >
              <step refunit='book' delim='.'
                    from='CHILD (1 DIV1 N %1)' to='DITTO' >
              <step refunit='poem' delim='.'
                    from='CHILD (1 DIV2 N %1)' to='DITTO' >
              <step refunit='line'
                    from='CHILD (1 L N %1)' to='DITTO' >
         </refsDecl>
         ...
    </encodingDesc>
</teiHeader>
```

The second example encodes the same reference system, again using the n attribute on the <div0>, <div1>, <div2>, and <l> tags, but giving the reference string in full on each tag. If canonical references are made only to lines, the reference system could be declared as follows:

```
 <refsDecl>
   <step refunit='line'
         from='DESCENDANT (1 L N %1)' to='DITTO' >
 </refsDecl>
```

Since no delimiter is specified, the entire canonical reference string is sought as the value of the n attribute on an <l> element.

In order to handle references to works, books, and poems as well as to individual lines, the declaration for the reference system must be more complicated:

```
<refsDecl>
  <step from='DESCENDANT (1 (DIV[012]|L) N %1)' to='DITTO' >
</refsDecl>
```

This declaration indicates that the entire reference string must be sought as the value of the n attribute on a <div0>, <div1>, <div2>, or <l> element.

The third example encodes the same reference system, this time giving the entire reference string as the value of the id attribute on the relevant tags. The reference system declaration for such an encoding would be:

```
<refsDecl>
    <step from="ID (%1)" to="DITTO">
</refsDecl>
```

As in the previous example, no single value can be given for the refunit attribute in this declaration, as the single step handles references to works, books, and poems, as well as to lines. The type attribute on the <div0>, <div1>, and <div2> elements may be used, however, to indicate the type of the result returned from a match.

Reference systems recorded by means of milestone tags can also be declared; the following prose description could be used to declare the example given in section 6.9.3 ('Milestone Tags') on p. 158.

```
<refsDecl>
    <p>Standard references to work, book, poem, and line may be
    constructed from the <gi>milestone</gi> tags in the text.
</refsDecl>
```

Or in this way, using a formal declaration for this reference scheme derived from edition E1.

```
<refsDecl>
    <state delim=' ' unit=work ed='E1'>
    <state delim='.' unit=book ed='E1'>
    <state delim=':' unit=poem ed='E1'>
    <state unit=line ed='E1'>
</refsDecl>
```

This is synonymous with the following declaration using the <step> element:

```
<refsDecl>
    <step refunit='work' delim=' '
          from='DESCENDANT (1 MILESTONE EDITION E1 UNIT work N %1)'
          to=  'FOLLLOWING (1 MILESTONE EDITION E1 UNIT work)' >
    <step refunit='book' delim='.'
          from='DESCENDANT (1 MILESTONE EDITION E1 UNIT book N %2)'
          to=  'FOLLLOWING (1 MILESTONE EDITION E1 UNIT book)' >
    <step refunit='poem' delim='.'
          from='DESCENDANT (1 MILESTONE EDITION E1 UNIT poem N %3)'
          to=  'FOLLLOWING (1 MILESTONE EDITION E1 UNIT poem)' >
    <step refunit='line'
          from='DESCENDANT (1 MILESTONE EDITION E1 UNIT line N %4)'
          to=  'FOLLLOWING (1 MILESTONE EDITION E1 UNIT line)' >
</refsDecl>
```

6.10 Bibliographic Citations and References

Bibliographic references (that is, full descriptions of bibliographic items such as books, articles, films, broadcasts, songs, etc.) or pointers to them may appear at various places in a TEI text. They are required at several points within the TEI Header's source description, as discussed in section 5.2.7 ('The Source Description') on p. 90; they may also appear within the body of a text,

either singly, (for example within a footnote), or collected together in a list as a distinct part of a text.

In printed texts, the individual constituents of a bibliographic reference are conventionally marked off from each other and from the flow of text by such features as bracketing, italics, special punctuation conventions, underlining, etc. In electronic texts, such distinctions are also important, whether in order to produce acceptably formatted output or to facilitate intelligent retrieval processing,[8] quite apart from the need to distinguish the reference itself as a textual object with particular linguistic properties.

It should be emphasized that for references as for other textual features, the primary or sole consideration is not how the text should be formatted when it is printed. The distinctions permitted by the scheme outlined here may not necessarily be all that particular formatters or bibliographic styles require, although they should prove adequate to the needs of many such commonly used software systems.[9] The features distinguished and described below (in section 6.10.2 ('Components of Bibliographic References') on p. 166) constitute a set which has been useful for a wide range of bibliographic purposes and in many applications, and which moreover corresponds to a great extent with existing bibliographic and library cataloguing practice. For a fuller account of that practice as applied to electronic texts see section 5.2.7 ('The Source Description') on p. 90; for a brief mention of related library standards see section 5.7 ('Note for Library Cataloguers') on p. 116.

6.10.1 Elements of Bibliographic References

The following elements are used to mark individual bibliographic references as wholes, or in groups:

<bibl> contains a loosely-structured bibliographic citation of which the sub-components may or may not be explicitly tagged.

<biblStruct> contains a structured bibliographic citation, in which only bibliographic subelements appear and in a specified order.

<biblFull> contains a fully-structured bibliographic citation, in which all components of the TEI file description are present.

<listBibl> contains a list of bibliographic citations of any kind.

These elements all share a number of possible component sub-elements. For the <bibl> and <biblStruct> elements, exactly the same sub-elements are concerned, and they are described together in section 6.10.2 ('Components of Bibliographic References') on p. 166; for the <biblFull> element, the sub-elements concerned are fully described in section 5.2 ('The File Description') on p. 80.

Different levels of specific tagging may be appropriate in different situations. In some cases, it may be felt necessary to mark just the extent of the reference itself, with perhaps a few distinctions being made within it (for example, between the part of the reference which identifies a title or author and the rest). Such references, containing a mixture of text with specialized bibliographic elements, are regarded as <bibl> elements, and tagged accordingly. For example:

```
<p>A book which had a great influence on him
was <bibl>Tufte's <title>Envisioning
Information</title></bibl>, although he may
never have actually read it.
```

Indeed, some encoders may find it unnecessary to mark the bibliographic reference at all:

```
<p>A book which had a great influence on him
was Tufte's <title>Envisioning Information</title>,
although he may never have actually read it.
```

[8] For example, to distinguish 'London' as an author's name from 'London' as a place of publication or as a component of a title.

[9] Among the bibliographic software systems and subsystems consulted in the design of the <biblStruct> structure were BibTeX, Scribe, and ProCite. The distinctions made by all three may be preserved in <biblStruct> structures, though the nature of their design prevents a simple one-to-one mapping from their data elements to TEI elements. For further information, see section 6.10.4 ('Relationship to Other Bibliographic Schemes') on p. 175.

Some bibliographic references are extremely elliptical, often only a string of the form 'Baxter, 1983'. If no further details of Baxter's book are given in the source text and none are supplied by the encoder, then the reference thus given should be tagged as a <bibl>:

```
All of this is of course much more fully treated
in <bibl>Baxter, 1983</bibl>.
```

In general, however, normal modern bibliographic practice, and these Guidelines, distinguish between a bibliographic *reference*,

which is a self-sufficient description of a bibliographic item, and a bibliographic *pointer*,

which is a short-form citation (such as 'Baxter, 1983') which serves usually as a place-holder or pointer to a full long-form reference found elsewhere in the text. The usual encoding of short-form references such as 'Baxter, 1983' is not as <bibl> elements but as cross-references to such elements; see section 6.10.3 ('Bibliographic Pointers ') on p. 175 below.

In cases where the encoder wishes to impose more structure on the bibliographic information, for example to make sure it conforms to a particular style-sheet or retrieval processor, the <biblStruct> element should be used. Note that several of the features in this and later examples are explained later in the current section.

```
<biblStruct>
    <monogr>
    <author>Edward R. Tufte</>
    <title>Envisioning Information</>
    <imprint>
         <pubPlace>Cheshire, Conn.</>
         <publisher>Graphics Press</>
         <date>1990</>
    </imprint></monogr>
</biblStruct>
```

The highest level of detail and the most complex structure supported by the current proposals is provided by the <biblFull> element, which closely resembles the <fileDesc> element of the TEI Header (section 5.2 ('The File Description') on p. 80).

```
<biblFull>
    <titleStmt>
         <title>Envisioning Information</>
         <author>Tufte, Edward R[olf]</>
      </titleStmt>
    <extent>126 pp.</>
    <publicationStmt>
         <creationDate>1990</>
         <publisher>Graphics Press</>
         <pubPlace>Cheshire, Conn. USA</>
         <date>1990</>
    </publicationStmt>
</biblFull>
```

A list of bibliographic items, of whatever kind, may be treated in the same way as any other list (see section 6.7 ('Lists') on p. 149). Alternatively, the specialized <listBibl> element may be used. The difference between the two is that a <list> contains <item> elements, within which bibliographic elements (<bibl>, <biblStruct> or <biblFull>) may appear, as well as other phrase- and paragraph-level elements, whereas the <listBibl> may contain only bibliographic elements, optionally preceded by a heading and a series of introductory paragraphs. The former would be appropriate for a list of bibliographic elements in which descriptive prose predominated, and the latter for a more formal bibliography. The following are thus both legal encodings of a list of bibliographic entries: a <listBibl>:

```
<listBibl>
<head>Bibliography</head>
<biblStruct id=NEL80>
    <analytic>
         <author>Nelson, T. H.</>
```

```
          <title>Replacing the printed word:
                 a complete literary system.</>
     </analytic>
     <monogr>
          <title>Information Processing '80:  Proceedings of the IFIPS
                 Congress, October 1980</>
          <editor>Simon H. Lavington</>
          <imprint>
               <publisher>North-Holland</>
               <pubPlace>Amsterdam</>
               <date>1980</>
          </imprint>
          <biblScope>pp 1013-23
     </monogr>
     <note>Apparently a draft of section 4 of <title>Literary
           Machines</title>.</note>
</biblStruct>
<bibl id=NEL88>Ted Nelson:  <title>Literary Machines</title>
     (privately published, 1987)</bibl>
<bibl id=BAX88>
     <author>Baxter, Glen</author>
     <title>Glen Baxter His Life: the years of struggle</title>
     London: Thames and Hudson, 1988.
     </bibl>
</listBibl>
```

or a simple <list>:

```
<list><head>Bibliography</>
<item><bibl id=NEL80>
        <author>Nelson, T. H.</>
        <title level=a>Replacing the printed word:
             a complete literary system.</>
        <title level=m>Information Processing '80:
             Proceedings of the IFIPS Congress, October 1980</>
        <editor>Simon H. Lavington</>
        <publisher>North-Holland</>
        <pubPlace>Amsterdam</>
        <date>1980</>
        <biblScope>pp 1013-23
        <note>Apparently a draft of section 4 of <title>Literary
         Machines</title>.</note>
     </bibl></item>
<item><bibl id=NEL88>Ted Nelson:  <title>Literary Machines</title>
     (privately published, 1987)</bibl></item>
<item><bibl id=BAX88>
     <author>Baxter, Glen</author>
     <title>Glen Baxter His Life: the years of struggle</title>
     London: Thames and Hudson, 1988.
     </bibl></item>
</list>
```

The formal declarations for these elements are as follows:

```
<!-- 6.10.1:  Tags for Bibliographic References              -->
<!ELEMENT bibl          - o  (#PCDATA | %m.phrase; |
                              %m.biblPart;)*                  >
<!ATTLIST bibl                %a.global;
                              %a.declarable;                  >
<!ELEMENT biblStruct    - o  (analytic?, (monogr, series*)+,
                              (note | idno)*)                 >
<!ATTLIST biblStruct          %a.global;
                              %a.declarable;                  >
```

```
<!ELEMENT biblFull      - o  (titleStmt, editionStmt?, extent?,
                              publicationStmt, seriesStmt?,
                              notesStmt?, sourceDesc*)          >
<!ATTLIST biblFull            %a.global;
                              %a.declarable;                    >
<!ELEMENT listBibl      - -  (head?, (bibl | biblStruct |
                              biblFull)+, trailer?)             >
<!ATTLIST listBibl            %a.global;
                              %a.declarable;                    >
<!-- (continued in sec. 6.10.2.1, 6.10.2.2, 6.10.2.3)         -->
<!-- This fragment is used in sec. 6.12                        -->
```

6.10.2 Components of Bibliographic References

This section discusses a number of very commonly occurring component elements of bibliographic references. They fall into four groups:

- elements for grouping components of the *analytic*, *monographic*, and *series* levels in a structured bibliographic reference
- titles of various kinds, and statements of intellectual responsibility (authorship, etc.)
- information relating to the publication, pagination, etc. of an item
- annotation, commentary and further detail

The following sections describe the elements which may be used to represent such information within a <bibl> or <biblStruct> element. Within the former, any or all of these may be used and in any order. Within the latter, such of these elements as exist for a given reference must be distinguished, and must also be presented in a specific order, discussed further below (section 6.10.2.6 ('Order of Components within References') on p. 175).

6.10.2.1 Analytic, Monographic, and Series Levels

In common library practice a clear distinction is made between an individual item within a larger collection and a free-standing book, journal, or collection. Similarly a book in a series is distinguished sharply from the series within which it appears. An article forming part of a collection which itself appears in a series thus has a bibliographic description with three quite distinct levels of information:

1. the *analytic* level, giving the title, author, etc., of the article;
2. the *monographic* level, giving the title, editor, etc., of the collection;
3. the *series* level, giving the title of the series, possibly the names of its editors, etc., and the number of the volume within that series.

In the same way, an article in a journal requires at least two levels of information: the analytic level describing the article itself, and the monographic level describing the journal.

These three levels may be distinguished within a <bibl> element, and must be distinguished within a <biblStruct> element if present, by means of the following tags:

<analytic> contains bibliographic elements describing an item (e.g. an article or poem) published within a monograph or journal and not as an independent publication.

<monogr> contains bibliographic elements describing an item (e.g. a book or journal) published as an independent item (i.e. as a separate physical object).

<series> contains information about the series in which a book or other bibliographic item has appeared.

For purposes of TEI encoding, journals and anthologies are both treated as monographs; a journal title will thus be tagged <title level=j> or <monogr><title> ... </title> ... </monogr>. Individual articles in the journal or collected texts should be treated at the "analytic" level. When an article has been printed in more than one journal or collection, the bibliographic reference may have more than one <monogr> element, each possibly followed by one or more <series> elements. A <series> element always relates to the most recently preceding <monogr> element. (Whether reprints of an article are treated in the same bibliographic reference or a separate one varies among different styles. Library lists typically use a different

entry for each publication, while academic footnoting practice typically treats all publications of the same article in a single entry.)

For example, the article cited in this example has been published twice, once in a journal and once in a collection which appeared in a series:

```
<biblStruct>
     <analytic>
          <author>Thaller, Manfred</author>
          <title level=a>A Draft Proposal for a Standard for the
                 Coding of Machine Readable Sources</>
     </analytic>
     <monogr>
          <!-- In -->
          <title level=j>Historical Social Research</title>
          <imprint>
                <biblScope type=vol>40</>
                <date>October 1986</>
                <biblScope type=pages>3-46</>
          </imprint>
     </monogr>
     <monogr>
          <!-- Rpt. in -->
          <title level=m>Modelling Historical Data:
                 Towards a Standard for Encoding and Exchanging
                 Machine-Readable Texts</title>
          <editor>Daniel I. Greenstein</editor>
          <imprint>
               <pubPlace>St. Katharinen</pubPlace>
               <publisher>Max-Planck-Institut f&uuml;r Geschichte
                     In Kommission bei
                     Scripta Mercaturae Verlag</publisher>
               <date>1991</date>
          </imprint>
     </monogr>
     <series>
          <title level=s>Halbgraue Reihe
               zur Historischen Fachinformatik</title>
          <respStmt><resp>Herausgegeben von</resp>
                <name type=person>Manfred Thaller</>
                <name type=org>Max-Planck-Institut f&uuml;r
                    Geschichte</>
          </respStmt>
          <title level=s>Serie A:  Historische Quellenkunden</>
          <biblScope>Band 11</biblScope>
     </series>
</biblStruct>
```

Punctuation may not appear between the elements within a structured bibliographic entry; if punctuation is to be given explicitly in the encoding, it must be contained within the elements it delimits. As the example shows, it is possible to encode the entry without any inter-element punctuation: this facilitates use of the <biblStruct> element in systems which can render bibliographic references in any of several styles.

The formal declarations for the elements defined in this section are as follows:

```
<!-- 6.10.2.1:  Tags for Bibliographic References (cont'd)    -->
<!-- (continuation of sec. 6.10.1)                            -->
<!ELEMENT analytic      - O  (author | editor | respStmt |
                              title)*                          >
<!ATTLIST analytic           %a.global;                        >
<!ELEMENT monogr        - O  ( ( ((author | editor |
                              respStmt)+, title+, (editor |
                              respStmt)*) | (title+, (author |
```

```
                              editor | respStmt)*))?, (note |
                              meeting)*, (edition, (editor |
                              respStmt)*)*, imprint, (imprint |
                              extent | biblScope)* )            >
<!ATTLIST monogr              %a.global;                        >
<!ELEMENT series        - O   (title | editor | respStmt |
                              biblScope)*                       >
<!ATTLIST series              %a.global;                        >
```

6.10.2.2 Authors, Titles, and Editors

Bibliographic references typically begin with a statement of the title being cited and the names of those intellectually responsible for it. For articles in journals or collections, such statements should appear both for the analytic and for the monographic level. The following elements are provided for tagging such elements:

<title> contains the title of a work, whether article, book, journal, or series, including any alternative titles or subtitles. Attributes include:

level indicates whether this is the title of an article, book, journal, series, or unpublished material. Legal values are:

a analytic title (article, poem, or other item published as part of a larger item)

m monographic title (book, collection, or other item published as a distinct item, including single volumes of multi-volume works)

j journal title

s series title

u title of unpublished material (including theses and dissertations unless published by a commercial press)

type classifies the title according to some convenient typology. Sample values include:

main main title

subordinate subtitle, title of part

parallel alternate title, often in another language, by which the work is also known

abbreviated abbreviated form of title

<author> in a bibliographic reference, contains the name of the author(s), personal or corporate, of a work; the primary *statement of responsibility* for any bibliographic item.

<editor> secondary *statement of responsibility* for a bibliographic item, for example the name of an individual, institution or organization, (or of several such) acting as editor, compiler, translator, etc.

<respStmt> supplies a statement of responsibility for someone responsible for the intellectual content of a text, edition, recording, or series, where the specialized elements for authors, editors, etc. do not suffice or do not apply.

<resp> contains a phrase describing the nature of a person's intellectual responsibility.

<name> contains a proper noun or noun phrase.

<meeting> in bibliographic references, contains a description of the meeting or conference from which the bibliographic item derives.

In bibliographic references, all titles should be tagged as such, whether analytic, monographic, or series titles. The single element <title> is used for all these cases. When it appears directly within an <analytic>, <monogr>, or <series> element, <title> is interpreted as belonging to the appropriate level. When it appears elsewhere, its level attribute should be used to signal its bibliographic level. It is a semantic error to give a value for the level attribute which is inconsistent with the context; such values may be ignored. The level value a implies the analytic level; the values m, j, and u imply the monographic level; the value s implies the series level. Note, however, that the semantic error occurs only if the nested title is directly enclosed by the <analytic>, <monogr>, or <series> element; if it is enclosed only indirectly, no semantic error need be present. For example, the analytic title may contain a monographic title:

```
<biblStruct>
    <analytic><author>Lucy Allen Paton</>
              <title>Notes on Manuscripts
                   of the <title level=m>Proph&eacute;cies
                   de Merlin</title></title>
    </analytic>
    <monogr><title level=j>PMLA</>
          <imprint><biblscope type=vol>8</>
            <date>1913</date>
            <biblscope type=pages>122</>
      </imprint>
    </monogr>
</biblStruct>
```

In this case, the analytic title "Notes on Manuscripts of the *Prophécies de Merlin*" needs no level attribute because it is directly contained by the <analytic> level; the monographic title contained within it, "Prophécies de Merlin," does not create a semantic error because it is not directly contained by the <analytic> element.

In some bibliographic applications, it may prove useful to distinguish main titles from subordinate titles, parallel titles, etc. The type attribute is provided to allow this distinction to be recorded.

The following reference, from a national standard for bibliographic references,[10] illustrates this type of analysis with its distinction between main and subordinate titles.

```
<bibl>Saarikoski, Pirkko-Liisa, and Paavo Suomalainen,
<title level=a type=main>Studies on the physiology of
the hibernating hedgehog, 15</title>
<title level=a type=subordinate>Effects of seasonal
and temperature changes on the in vitro
glycerol release from brown adipose tissue</>
<title level=j>Ann. Acad. Sci. Fenn., Ser. A4</>
<date>1972</> <biblScope type='vol: pp'>187: 1-4</>
</bibl>
```

Slightly more complex is the distinction made below among main, subordinate, and parallel titles, in an example from the same source (p. 63). The punctuation and the bibliographic analysis are those given in ANSI Z39.29-1977; the punctuation is in the style prescribed by the International Standard Bibliographic Description (ISBD).[11]

```
<bibl>Tchaikovsky, Peter Ilich.
<title level=m type=main>The swan lake ballet</>
= <title level=m type=parallel>Le lac des cygnes</>
: <title level=m type=subordinate>grand ballet en 4 actes</>
: <title level=m type=subordinate>op. 20</>
[Score].
New York:  Broude Brothers; [1951] (B.B. 59). vi, 685 p.
</bibl>
```

The elements <author> and <editor> have, for printed books and articles, a fairly obvious significance; for other kinds of bibliographic items their proper usage may be less obvious. The <author> element should be used for the person or agency with primary responsibility for a work's intellectual content, and the element <editor> for an editor of the work. Thus an organization such as a radio or television station is usually accounted "author" of a broadcast, for example, while the author of a Government report will usually be the agency which produced it.

For anyone else with responsibility for the work, the <respStmt> element should be used. The nature of the responsibility is indicated by means of a <resp> element, and the person,

[10] American National Standard for Bibliographic References, ANSI Z39.29-1977 (New York: American National Standards Institute, 1977), p. 34 (sec. A.2.2.1).

[11] The analysis is not wholly unproblematic: as the text of the standard points out, the first subordinate title is subordinate only to the parallel title in French, while the second is subordinate to both the English main title and the French parallel title, without this relationship being made clear, either in the markup given in the example or in the reference structure offered by the standard.

organization etc. responsible by a <name> element. At least one of each of these must be given within the <respStmt> element, followed optionally by any number of either. Examples of secondary responsibility of this kind include the roles of illustrator, translator, editor, annotator. The <respStmt> element may also be used for editors, if it is desired to record the specific terms in which their role is described.

Examples of <author> and <editor> may be found in sections 6.10.1 ('Elements of Bibliographic References') on p. 163, and 6.10.2.1 ('Analytic, Monographic, and Series Levels') on p. 166; wherever <author> and <editor> may occur, the <respStmt> element may also occur. When one of these elements precedes or immediately follows a title, it applies to that title; when it follows an <edition> element or occurs within an edition statement, it applies to the edition in question.

In this example, the <respStmt> elements apply to the work as a whole, not merely to the first edition:

```
<bibl>
   <author>Lominadze, D. G.</>
   <title level=m>Cyclotron waves in plasma.</title>
   <respStmt><resp>translated by</> <name>A. N. Dellis;</name>
             <resp>edited by</> <name>S. M. Hamberger.</name>
   </respStmt>
   <edition>1st ed.</>
   <imprint><pubPlace>Oxford:</> <publisher>Pergamon Press,</>
      <date>1981.</></imprint>
   <extent>206 p.</>
   <title level=s>International series in natural philosophy.</>
   <note place=inline>Translation of:
     <title level=m lang=RU>Ciklotronnye volny v plazme.</></>
</bibl>
```

In this example, by contrast, the <respStmt> element applies to the edition, and not to the collection per se (Moser and Tervooren were not responsible for the first thirty-five printings); the elements of the reference have been reordered from their appearance on the title page of the volume in order to ensure the correct relationship of the collection title, the edition statement, and the statement of responsibility.

```
<biblStruct>
<monogr>
    <title>Des Minnesangs Fr&uuml;hling</title>
    <note place=inline>Mit 1 Faksimile</note>
    <edition>36., neugestaltete und erweiterte Auflage</edition>
    <respStmt>
         <resp>Unter Benutzung der Ausgaben
         von <name>Karl Lachmann</> und <name>Moriz Haupt</>,
         <name>Friedrich Vogt</> und <name>Carl von Kraus</>
         bearbeitet von
         </resp>
         <name>Hugo Moser</>
         <!-- und -->
         <name>Helmut Tervooren</>
    </respStmt>
    <imprint>
         <biblScope type=volume>I</>
         <biblScope type='volume title'>Texte</>
         <pubPlace>Stuttgart</>
         <publisher>S. Hirzel Verlag</>
         <date>1977</>
    </imprint>
</monogr>
</biblStruct>
```

With the exception of the <name> element (for which see section 6.4 ('Names, Numbers, Dates, Abbreviations, and Addresses') on p. 132), the elements described in this section are

defined as follows:

```
<!-- 6.10.2.2:  Tags for Bibliographic References (cont'd)    -->
<!-- (continuation of sec. 6.10.1)                            -->
<!ELEMENT author        - o  (%phrase.seq)                      >
<!ATTLIST author             %a.global;                         >
<!ELEMENT editor        - o  (%phrase.seq)                      >
<!ATTLIST editor             %a.global;
          role               CDATA               editor       >
<!ELEMENT respStmt      - o  ((resp & name), (resp | name)*)    >
<!ATTLIST respStmt           %a.global;                         >
<!ELEMENT resp          - o  (%phrase.seq;)                     >
<!ATTLIST resp               %a.global;                         >
<!ELEMENT title         - o  (%paraContent)                     >
<!ATTLIST title              %a.global;
          level              (a | m | j | s | u) #IMPLIED
          type               CDATA               #IMPLIED     >
<!ELEMENT meeting       - -  (%paraContent)                     >
<!ATTLIST meeting            %a.global;                         >
```

6.10.2.3 Imprint, Pagination, and Other Details

By 'imprint' is meant all the information relating to the publication of a work: the person or organization by whose authority and in whose name a bibliographic entity such as a book is made public or distributed (whether a commercial publisher or some other organization), the place of publication, and a date. It may also include a full address for the publisher or organization. Full bibliographic references usually specify either the number of pages in a print publication (or equivalent information for non-print materials), or the specific location of the material being cited within its containing publication. The following elements are provided to hold this information:

<imprint> groups information relating to the publication or distribution of a bibliographic item.

<address> contains a postal or other address, for example of a publisher, an organization, or an individual.

<pubPlace> contains the name of the place where a bibliographic item was published.

<publisher> provides the name of the organization responsible for the publication or distribution of a bibliographic item.

<date> contains a date in any format.

<idno> supplies any standard or non-standard number used to identify a bibliographic item. Attributes include:

- **type** categorizes the number, for example as an ISBN or other standard series.

<extent> describes the approximate size of the electronic text as stored on some carrier medium, specified in any convenient units.

<biblScope> defines the scope of a bibliographic reference, for example as a list of pagenumbers, or a named subdivision of a larger work. Attributes include:

- **type** identifies the type of information conveyed by the element, e.g. "pages", "volume". Suggested values include:
  - ***volume*** the element contains a volume number.
  - ***issue*** the element contains an issue number, or volume and issue numbers.
  - ***pages*** the element contains a page number or page range.
  - ***chapter*** the element contains a chapter indication (number and/or title)
  - ***part*** the element identifies a part of a book or collection.

For bibliographic purposes, usually only the place (or places) of publication are required, possibly including the name of the country, rather than a full address; the element <pubPlace> is provided for this purpose. Where however the full postal address is likely to be of importance in identifying or locating the bibliographic item concerned, it may be supplied and tagged using the <address> element described in section 6.4.2 ('Addresses') on p. 134. Alternatively, if desired, the <rs> or <name> elements described in section 6.4.1 ('Referring Strings') on p. 132 may be

used; this involves no claim that the information given is either a full address or the name of a city.

The name of the publisher of an item should be marked using the <publisher> tag even if the item is made public ("published") by an organization other than a conventional publisher, as is frequently the case with technical reports:

```
<biblStruct>
<monogr>
    <author>Nicholas, Charles K.</>
    <author>Welsch, Lawrence A.</>
    <title>On the interchangeability of SGML and ODA</>
    <imprint>
         <pubPlace>Gaithersburg, MD</pubPlace>
         <publisher>National Institute of Standards and Technology</>
         <date value='1992-01'>January 1992</>
    </imprint>
</monogr>
<extent>19 pp.</>
<idno type='NIST'>NISTIR 4681</>
</biblStruct>
```

and with dissertations:

```
<biblStruct>
<monogr>
    <author>Hansen, W.</>
    <title level=u>Creation of hierarchic text
         with a computer display</>
    <note place=inline>Ph.D. dissertation</>
    <imprint>
         <publisher>Dept. of Computer Science, Stanford Univ.</>
         <pubPlace>Stanford, CA</>
         <date value='1971-06'>June 1971</>
    </imprint>
</monogr>
</biblStruct>
```

When an item has been reprinted, especially reprinted without change from a specific earlier edition, the reprint may appear in a <monogr> element with only the <imprint> and other details of the reprint. In the following example, a microform reprint has been issued without any change in the title or authorship. The series statement here applies only to the second <monogr> element.

```
<biblStruct>
<monogr>
    <author>Shirley, James</>
    <title type=main>The gentlemen of Venice</>
    <title type=subordinate>a tragi-comedie presented at
         the private house in Salisbury Court by
         Her Majesties servants</>
    <note place=inline>[Microform]</>
    <imprint>
         <pubPlace>London</>
         <publisher>H. Moseley</>
         <date>1655</>
    </imprint>
    <extent>78 p.</>
</monogr>
<monogr>
    <imprint>
         <pubPlace>New York</>
         <publisher>Readex Microprint</>
         <date>1953</>
```

```
    </imprint>
   <extent>1 microprint card, 23 x 15 cm.</>
</monogr>
<series>
    <title>Three centuries of drama:  English, 1642-1700</>
</series>
 </biblStruct>
```

A bibliographic description, particularly for an analytic title, will often include some additional information specifying its location, for example as a volume number, page number, range of page numbers, or name or number of a subdivision of the host work. The element <biblScope> may be used to identify such information if it is present. Where it is desired to distinguish different classes of such information (volume number, page number, chapter number, etc.), the type attribute may be used with any convenient typology.

When the item being cited is a journal article, the <imprint> element describing the issue in which it appeared will typically contain <biblScope> elements for volume and page numbers, together with a <date> element.

For example:

```
<biblStruct>
<analytic>
    <author>Wrigley, E. A.</>
    <title>Parish registers and the historian</>
</analytic>
<!-- in -->
<monogr>
    <editor>Steel, D. J.</>
    <title>National index of parish registers</>
    <imprint>
         <pubPlace>London</pubPlace>
         <publisher>Society of Genealogists</>
         <date value='1968'>1968</>
    </imprint>
    <biblScope type=volume>vol. 1</>
    <biblScope type=pages>pp. 155-167.</>
</monogr>
</biblStruct>
```

The type attribute on <biblScope> is optional: both the following are legal examples:

```
<biblStruct>
<analytic>
    <author>Boguraev, Branimir</>
    <author>Neff, Mary</>
    <title>Text Representation, Dictionary Structure,
         and Lexical Knowledge</>
</analytic>
<!-- in -->
<monogr>
    <title level=j>Literary & Linguistic Computing</>
    <imprint>
         <biblScope type=volume>7</>
         <biblScope type=issue>2</>
         <date>1992</>
         <biblScope type=pages>110-112</>
    </imprint>
</monogr>
</biblStruct>
```

```
<biblStruct>
<analytic>
    <author>Chesnutt, David</>
    <title>Historical Editions in the States</>
```

```
</analytic>
<!-- in -->
<monogr>
    <title level=j>Computers and the Humanities</>
    <imprint>
         <biblScope>25.6</>
         <date value='1991-12'>(December, 1991):</>
         <biblScope>377-380</>
    </imprint>
</monogr>
</biblStruct>
```

Formal definitions for the elements described in this section are as follows:

```
<!-- 6.10.2.3:  Tags for Bibliographic References (cont'd)    -->
<!-- (continuation of sec. 6.10.1)                            -->
<!ELEMENT imprint       - O  (pubPlace | publisher | date |
                              biblScope)*                       >
<!ATTLIST imprint            %a.global;                         >
<!ELEMENT publisher     - o  (%phrase.seq)                      >
<!ATTLIST publisher          %a.global;                         >
<!ELEMENT biblScope     - o  (%phrase.seq)                      >
<!ATTLIST biblScope          %a.global;
          type               CDATA               #IMPLIED      >
<!ELEMENT pubPlace      - -  (%phrase.seq;)                     >
<!ATTLIST pubPlace           %a.global;
                             %a.names;                          >

<!-- Note and date are defined elsewhere, as are extent,      -->
<!-- address, and idno.                                       -->
```

6.10.2.4 Series Information

Series information may (in <bibl> elements) or must (in <biblStruct> elements) be enclosed in a <series> element or (in a <biblFull> element) a <seriesStmt> element. The title of the series may be tagged <title level=s>, the volume number <biblScope type=volume>, and responsibility statements for the series (e.g. the name and affiliation of the editor, as in the example in section 6.10.2.1 ('Analytic, Monographic, and Series Levels') on p. 166) may be tagged <editor> or <respStmt>.

6.10.2.5 Notes and Other Additional Information

Explanatory notes about the publication of unusual items, the form of an item (e.g. '[Score]' or '[Microform]'), or its provenance (e.g. 'translation of ...') may be tagged using the <note> element. The same element may be used for any descriptive annotation of a bibliographic entry in a database.

<note> contains a note or annotation. Attributes include:

type describes the type of note.

place indicates where the note appears in the source text. Sample values include:

foot note appears at foot of page.

end note appears at end of chapter or volume.

inline note appears as a marked paragraph in the body of the text.

left note appears in left margin.

right note appears in right margin.

interlinear note appears between lines of the text.

app[aratus] note appears in the apparatus at the foot of the page.

For example:

```
<bibl><author>Coombs, James H., Allen H. Renear,
       and Steven J. DeRose.</author>
       <title level=a>Markup Systems and the Future of Scholarly
       Text Processing.</title>
       <title level=j>Communications of the ACM</title>
       <biblScope>30.11 (November 1987):  933-947.</biblScope>
       <note>Classic polemic supporting descriptive over procedural
            markup in scholarly work.</note>
```

6.10.2.6 Order of Components within References

The order of elements in <bibl> elements is not constrained.

In <biblStruct> elements, the <analytic> element, if it occurs, must come first, followed by one or more <monogr> and <series> elements, which may appear intermingled (as long as a <monogr> element comes first). Within <analytic>, the title(s), author(s), editor(s), and other statements of responsibility may appear in any order; it is recommended that all forms of the title be given together. Within <monogr>, the author, editor, and statements of responsibility may either come first or else follow the monographic title(s). Following these, the elements must appear in the following order:

- <note>s on the publication (and <meeting> elements describing the conference, in the case of a proceedings volume)
- <edition> elements, each followed by any related <editor> or <respStmt> elements
- <imprint>
- <biblScope>

Within <imprint>, the elements allowed may appear in any order. Finally, within the <series> information in a <biblStruct>, the sequence of elements is not constrained.

If more detailed structuring of a bibliographic description is required, the <biblFull> element should be used. This is not further described here, as its contents are essentially equivalent to those of the <fileDesc> element in the <teiHeader>, which is fully described in section 5.2 ('The File Description') on p. 80.

6.10.3 Bibliographic Pointers

References which are pointers to bibliographic items, of whatever kind, should be treated in the same way as other cross-references (see section 6.6 ('Simple Links and Cross References') on p. 147). As discussed in that section, cross referencing within TEI texts is in general represented by means of <ptr> or <ref> elements. A target attribute on these elements is used to supply an identifying value for the target of the cross reference, which should be, in the case of bibliographic elements, a bibliographic reference of some kind. Where the form of the reference itself is unimportant, or may be reconstructed mechanically, or is not to be encoded, the <ptr> element is used, as in the following example:

```
As shown above (<ptr target=NEL80>)...
```

Where the form of the reference is important, or contains additional qualifying information which is to be kept but distinguished from the surrounding text, the <ref> element should be used, as in the following example:

```
Nelson claims <ref target=NEL80>(ibid, passim)</ref> ...
```

It may be important to distinguish between the short form of a bibliographic reference and some qualifying or additional information. The latter should not appear within the scope of the <ref> element when this is the case, as for example in an application concerned to normalize bibliographic references:

```
Nelson claims (<ref target=NEL80>Nelson [1980]</ref>, pages 13-37) ...
```

6.10.4 Relationship to Other Bibliographic Schemes

The bibliographic tagging defined here can capture the distinctions required by most bibliographic encoding systems; for the benefit of users of some commonly used systems, the following

lists of equivalences are offered, showing the relationship of the markup defined here to the fields defined for bibliographic records in the Scribe, BibTeX, and ProCite systems.

The various bibliographic fields defined for use in the Scribe and BibTeX systems of bibliographic databases have the following equivalents in the scheme presented here:[12] Elements and structures available in the tag set defined here which have no analogues in Scribe and BibTeX are not noted.

address tag as <city>, <place>, or <address>
annote tag as <note>
author tag as <author>
booktitle tag as <title level=m> or <title> within <monogr>
chapter tag as <biblScope type=chapter>
date used only to record date entry was made in the bibliographic database; not supported
edition tag as <edition>
editor tag as <editor> or <respStmt>
editors tag as multiple <editor> or <respStmt> elements
fullauthor use the reg attribute on <author> or <name>
fullorganization use the reg attribute on <name type=org>
howpublished tag as <note>, possibly using the form <note place=inline>
institution used only for issuer of technical reports; tag as <publisher>
journal tag as <title level=j> or <title> within <monogr>
key used to specify an alternate sort key for the bibliographic item, for use instead of author's or editor's name; not supported
meeting tag as <meeting> or as <note>
month use <date>; if the date is not in a trivially parseable form, use the value attribute to provide an ISO-format equivalent
note tag as <note>
number tag as <biblScope type=issue> or <biblScope type=number>; for technical report numbers, use <idno type=docno>
organization used only for sponsor of conference; use <name type=org> within <respStmt> within <meeting> element
pages tag as <biblScope type=pages>
publisher tag as <publisher>
school used only for institutions at which thesis work is done; tag as <publisher>
series tag as <title level=s> or <title> within <series>
title tag as <title> in appropriate context or with appropriate level value
volume tag as <biblScope type=volume>
year tag as <date>; if the date is not in a trivially parseable form, use the value attribute to provide an ISO-format equivalent

6.11 Passages of Verse or Drama

The following elements are included in the core tag set for the convenience of those encoding texts which include mixtures of prose, verse and drama.

<l> contains a single, possibly incomplete, line of verse. Attributes include:

part specifies whether or not the line is metrically complete. Legal values are:

Y the line is metrically incomplete
N either the line is complete, or no claim is made as to its completeness
I the initial part of an incomplete line
M a medial part of an incomplete line
F the final part of an incomplete line

[12] The BibTeX scheme is intentionally compatible with that of Scribe, although it omits some fields used by Scribe. Hence only one list of fields is given here.

<lg> contains a group of verse lines functioning as a formal unit e.g. a stanza, refrain, verse paragraph, etc.

<sp> An individual speech in a performance text, or a passage presented as such in a prose or verse text.

<speaker> A specialized form of heading or label, giving the name of one or more speakers in a dramatic text or fragment.

<stage> contains any kind of stage direction within a dramatic text or fragment. Attributes include:

type indicates the kind of stage direction. Suggested values include:

setting describes a setting.

entrance describes an entrance.

exit describes an exit.

business describes stage business.

novelistic is a narrative, motivating stage direction.

delivery describes how a character speaks.

modifier gives some detail about a character.

location describes a location.

mixed more than one of the above

Full details of other, more specialized, elements for the encoding of texts which are predominantly verse or drama are described in the appropriate chapter of part three (for verse, see the verse base described in chapter 9 ('Base Tag Set for Verse ') on p. 213; for performance texts, see the drama base described in chapter 10 ('Base Tag Set for Drama') on p. 227). In this section, we describe only the elements listed above, all of which can appear in any text, whichever of the three modes prose, verse, or drama may predominate in it.

6.11.1 Core Tags for Verse

Like other written texts, verse texts or poems may be hierarchically subdivided, for example into books or cantos. These structural subdivisions should be encoded using the general purpose or <div1> (etc.) elements described below in chapters 8 ('Base Tag Set for Prose') on p. 211 and 9 ('Base Tag Set for Verse ') on p. 213. The fundamental unit of a verse text is the verse line rather than the paragraph, however.

The <l> element is used to mark up verse lines, that is metrical rather than typographic lines. Where a metrical line is interrupted by a typographic line break, the encoder may choose to ignore the fact entirely or to use the empty <lb> (line break) element discussed in 6.9 ('Reference Systems') on p. 155. In the copy text, the following example is printed on four typographic lines, beginning with the words 'There', 'From', 'The', and 'the'.

```
<l>There they lie, in the largest, in an
   open space in the woods,
<l>From 500 to 600 poor fellows &mdash; the groans
   and screams &mdash;
<l>The odor of blood, mixed with the fresh scent
   of the night, <lb>the grass, the trees &mdash;
   that Slaughter-house!
```

Where verse lines are not properly nested within the enclosing hierarchy (for example where verse lines cross larger boundaries such as verse paragraphs or speeches) the encoder may choose to use one of the techniques discussed in chapter 31 ('Multiple Hierarchies') on p. 633, or to use the part attribute to indicate that the verse line is incomplete, as in the following example:

```
<l>Thou fumblest <name>Eros</>, and my Queenes a Squire
<l>More tight at this, then thou:  Dispatch. O Loue,
<l>That thou couldst see my Warres to day, and knew'st
<l>The Royall Occupation, thou should'st see
<l part=i>A Workeman in't.
<stage>Enter an Armed Soldier.</stage>
<l part=f>Good morrow to thee, welcome. ...
```

In some verse forms, regular groupings of lines are regarded as units of some kind, often identified by a regular verse scheme. In stichic verse and couplets, groups of lines analogous to paragraphs are often indicated by indentation. In other verse forms, lines are grouped into irregular sequences indicated simply by white space. The neutral <lg> or line group element may be used to mark any such grouping of lines; the type is available to further categorize the line group where this is felt desirable, as in the following example. This example also demonstrates the rend attribute to indicate whether or not a line is indented.

```
<lg type=stanza>
<l>Come fill up the Glass,
<l rend=indent>Round, round let it pass,
<l>'Till our Reason be lost in our Wine:
<l rend=indent>Leave Conscience's Rules
<l rend=indent>To Women and Fools,
<l>This only can make us divine.
</lg>
<lg type=refrain n='Chorus'>
<l>Then a Mohock, a Mohock I'll be,
<l>No Laws shall restrain
<l>Our Libertine Reign,
<l>We'll riot, drink on, and be free.
</lg>
```

For some kinds of analysis, it may be useful to identify different kinds of line group within the same piece of verse. Such line groups may self-nest, in much the same way as the un-numbered element described in chapter 8 ('Base Tag Set for Prose') on p. 211. For example:

The part attribute may also be attached to a <lg> element to indicate that it is incomplete, for example because it forms part of a group that is divided between two speakers, as in the following example:

```
<sp><speaker>First Voice</>
<lg type=stanza part=I>
<l>But why drives on that ship so fast
<l>Withouten wave or wind?
</lg>
<sp><speaker>Second Voice</speaker>
<lg part=F>
<l>The air is cut away before,
<l>And closes from behind.
</lg>
```

For alternative methods of aligning groups of lines which do not form simple hierarchic groups, or which are discontinuous, see the more detailed discussion in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331. For discussion of other elements and attributes specific to the encoding of verse, see chapter 9 ('Base Tag Set for Verse ') on p. 213.

These elements are defined as follows:

```
<!-- 6.11.1:  Verse                                              -->
<!ELEMENT l             - O  (%paraContent)                      >
<!ATTLIST l                  %a.global;
                             %a.enjamb;
                             %a.metrical;
          part               (Y | N | I | M | F) N                >
<!ELEMENT lg            - O  ((%m.divtop)*, (l | lg)+,
                             (%m.divbot)*)                       >
<!ATTLIST lg                 %a.global;
                             %a.divn;
                             %a.metrical;                        >
<!-- This fragment is used in sec. 6.12                          -->
```

6.11.2 Core Tags for Drama

Like other written texts, dramatic and other *performance texts* such as cinema or TV scripts are often hierarchically organized, for example into acts and scenes. These structural subdivisions should be encoded using the general purpose or <div1> (etc.) elements described below in chapters 8 ('Base Tag Set for Prose') on p. 211 and 10 ('Base Tag Set for Drama') on p. 227. Within these divisions, the body of a performance text typically consists of *speeches*, often prefixed by a phrase indicating who is speaking, and occasionally interspersed with stage directions of various kinds.

In the following simple example, each speech consists of a single paragraph:

```
<div2 type=scene n='I.2'>
<head>Scene 2.</head>
<stage type=setting>Peachum, Filch.</stage>
<sp><speaker>FILCH.</speaker><p>Sir, Black Moll hath sent word her
Trial comes on in the Afternoon, and she hopes you will order Matters
so as to bring her off.
<sp><speaker>PEACHUM.</speaker><p>Why, she may plead her Belly
at worst; to my Knowledge she hath taken care of that Security.
But, as the Wench is very active and industrious, you may satisfy
her that I'll soften the Evidence.
<sp><speaker>FILCH.</speaker><p>Tom Gagg, sir, is found guilty.
```

In the following example, each speech consists of a sequence of verse lines, some of them being marked as metrically incomplete:

```
<div1 type='Act' n='I'><head>ACT I</>
<div2 type='Scene' n='1'><head>SCENE I</head>
<stage rend=italic>
Enter Barnardo and Francisco, two Sentinels, at several doors</stage>
<sp><speaker>Barn</><l part=Y>Who's there?
<sp><speaker>Fran</><l>Nay, answer me.  Stand and unfold yourself.
<sp><speaker>Barn</><l part=i>Long live the King!
<sp><speaker>Fran</><l part=m>Barnardo?
<sp><speaker>Barn</><l part=f>He.
<sp><speaker>Fran</><l>You come most carefully upon your hour.
<sp><speaker>Barn</><l>'Tis now struck twelve.  Get thee to bed,
 Francisco.
<sp><speaker>Fran</><l>For this relief much thanks.'Tis bitter cold,
<l part=i>And I am sick at heart.
```

In some cases, as here in the First Quarto of *Hamlet*, the printed speaker attributions need to be supplemented by use of the who attribute; again, the lines are marked as complete or incomplete:

```
<stage>Enter two Centinels.
<add place=right resp=ms>
Now call'd Bernardo & Francesco.
</add></stage>
<sp who='Francisco'><speaker>1.</>
    <l part=y>STand:  who is that?</sp>
<sp who='Barnardo'><speaker>2.</>
    <l part=y>Tis I.</sp>
<sp who='Francisco'><speaker>1.</>
    <l>O you come most carefully vpon your
       watch,</sp>
<sp who='Barnardo'><speaker>2.</>
    <l>And if you meete Marcellus and Horatio,
    <l>The partners of my watch, bid them make haste.</sp>
<sp who='Francisco'><speaker>1.</>
    <l part=y>I will:  See who goes there.</sp>
<stage>Enter Horatio and Marcellus.</>
<sp who='Horatio'><speaker>Hor.</>
```

```
    <l part=i>Friends to this ground.</sp>
<sp who='Marcellus'><speaker>Mar.</>
    <l part=f>And leegemen to the Dane,
    <l>O farewell honest souldier, who hath
    releeued you?</sp>
<sp who='Francisco'><speaker>1.</>
    <l>Barnardo hath my place, giue you good night.</sp>
```

By contrast with the preceding examples, the following encodes an early printed edition without making any assumption about which parts are prose or verse:

```
<div1 type=act n='I'>
<div2 type=scene n='1'>
<head rend=italic>Actus primus, Scena prima.</head>
<stage type=setting rend=italic>
A tempestuous noise of Thunder and Lightning heard:  Enter
a Ship-master, and a Boteswaine.</stage>
<sp><speaker>Master.</speaker><p> Bote-swaine.</sp>
<sp><speaker>Botes.</speaker><p> Heere Master: What cheere?</sp>
<sp><speaker>Mast.</speaker><p> Good: Speake to th' Mariners: fall
too't, yarely, or we run our selues a ground,
bestirre, bestirre.  <stage type=move>Exit.</stage></sp>
<stage type=move>Enter Mariners.</stage>
<sp><speaker>Botes.</speaker>
<p>Heigh my hearts, cheerely, cheerely my harts:
yare, yare: Take in the toppe-sale: Tend to th' Masters
whistle: Blow till thou burst thy winde, if roome e-nough.</sp>
```

The <sp> and <stage> elements should also be used to mark parts of a text otherwise in prose which are presented as if they were dialogue in a play. For example:

```
<sp><speaker>The reverend Doctor Opimiam</speaker>
<p>I do not think I have named a single unpresentable
fish. </sp>
<sp><speaker>Mr Gryll</speaker><p>Bream, Doctor: there
is not much to be said for bream.</sp>
<sp><speaker>The Reverend Doctor
Opimiam</speaker><p>On the contrary, sir, I think
there is much to be said for him.  In the first
place....
<p>Fish, Miss Gryll -- I could discourse to you on
fish by the hour:  but for the present I will
forbear...
</sp>
<sp><speaker>Lord Curryfin</speaker>
<stage>(after a pause).</stage>
<p><q>Mass</q> as the second grave-digger
says in <title>Hamlet</title>,
<q>I cannot tell.</q>
</sp>
<p>A chorus of laughter dissolved the sitting.
```

These elements are defined as follows:

```
<!-- 6.11.2:  Drama                                          -->
<!ELEMENT sp           - O  (speaker?, (p | l | lg | seg |
                            stage)+)                        >
<!ATTLIST sp                %a.global;
          who               IDREFS              #IMPLIED      >
<!ELEMENT speaker      - O  (%phrase.seq)        -(speaker)    >
<!ATTLIST speaker           %a.global;                        >
<!ELEMENT stage        - -  (%specialPara)       -(stage)      >
<!ATTLIST stage             %a.global;
          type              CDATA               mix           >
<!-- This fragment is used in sec. 6.12                      -->
```

6.12 Overview of the Core Tag Set

Except for those tags designed to be used in concurrent markup streams, all the elements described in this chapter occur in the *core* of TEI tags, defined by the following DTD fragment.

```
<!-- 6.12: Elements available in all forms of the TEI main    -->
<!-- DTD                                                      -->
<!-- Definition of elements, sub-group by sub-group.          -->

<!-- ... declarations from section 6.1                        -->
<!--     (Paragraph)                                          -->
<!--     go here ...                                          -->
<!-- ... declarations from section 6.3.2.1                    -->
<!--     (Highlighted phrases)                                -->
<!--     go here ...                                          -->
<!-- ... declarations from section 6.4.1                      -->
<!--     (Proper Nouns)                                       -->
<!--     go here ...                                          -->
<!-- ... declarations from section 6.4.3                      -->
<!--     (Numbers and measures)                               -->
<!--     go here ...                                          -->
<!-- ... declarations from section 6.4.4                      -->
<!--     (Dates and times)                                    -->
<!--     go here ...                                          -->
<!-- ... declarations from section 6.4.5                      -->
<!--     (Abbreviations)                                      -->
<!--     go here ...                                          -->
<!-- ... declarations from section 6.5.1                      -->
<!--     (Editorial tags for correction)                      -->
<!--     go here ...                                          -->
<!-- ... declarations from section 6.5.2                      -->
<!--     (Editorial tags for regularization)                  -->
<!--     go here ...                                          -->
<!-- ... declarations from section 6.5.3                      -->
<!--     (Other editorial tags)                               -->
<!--     go here ...                                          -->
<!-- ... declarations from section 6.4.2                      -->
<!--     (Addresses and their components)                     -->
<!--     go here ...                                          -->
<!-- ... declarations from section 6.6                        -->
<!--     (Simple cross references)                            -->
<!--     go here ...                                          -->
<!-- ... declarations from section 6.7                        -->
<!--     (Lists and List Items)                               -->
<!--     go here ...                                          -->
<!-- ... declarations from section 6.8.1                      -->
<!--     (Annotation)                                         -->
<!--     go here ...                                          -->
<!-- ... declarations from section 6.9.3                      -->
<!--     (Milestone tags)                                     -->
<!--     go here ...                                          -->
<!-- ... declarations from section 6.10.1                     -->
<!--     (Tags for Bibliographic References)                  -->
<!--     go here ...                                          -->
<!-- ... declarations from section 6.11.1                     -->
<!--     (Verse)                                              -->
<!--     go here ...                                          -->
<!-- ... declarations from section 6.11.2                     -->
<!--     (Drama)                                              -->
<!--     go here ...                                          -->
```

Chapter 7

Default Text Structure

This chapter describes the default high level structure for all TEI documents. The majority of the different base tag sets described in part II simply embed the framework defined in this chapter while a few redefine it with some minor modifications. This chapter is therefore relevant to every kind of TEI document. For further details on the overall structure of the TEI document type definitions, in particular the use of base and additional tag sets, see chapter 3 ('Structure of the TEI Document Type Definition') on p. 35.

TEI texts may be regarded either as *unitary*, that is, forming an organic whole, or as *composite,* that is, consisting of several components which are in some important sense independent of each other. The distinction is not always entirely obvious: for example a collection of essays might be regarded as a single item in some circumstances, or as a number of distinct items in others. In such borderline cases, the encoder must choose whether to treat the text as unitary or composite; each may have advantages and disadvantages in a given situation.

Whether unitary or composite, the text is marked with the **<text>** tag and may contain front matter, a text body, and back matter. In unitary texts, the text body is tagged **<body>**; in composite texts, where the text body consists of a series of subordinate texts or groups, it is tagged **<group>**. The overall structure of any text, unitary or composite, is thus defined by the following elements:

<text> contains a single text of any kind, whether unitary or composite, for example a poem or drama, a collection of essays, a novel, a dictionary, or a corpus sample.

<front> contains any prefatory matter (headers, title page, prefaces, dedications, etc.) found before the start of a text proper.

<body> contains the whole body of a single unitary text, excluding any front or back matter.

<group> contains the body of a composite text, grouping together a sequence of distinct texts (or groups of such texts) which are regarded as a unit for some purpose, for example the collected works of an author, a sequence of prose essays, etc.

<back> contains any appendixes, etc. following the main part of a text.

The overall structure of a unitary text is:

```
<TEI.2>
<TeiHeader> ... </TeiHeader>
<text>
    <front>
       <!-- front matter of copy text goes here. -->
    </front>
    <body>
       <!-- body of text goes here. -->
    </body>
    <back>
       <!-- back matter of text, if any, here. -->
    </back>
```

```
</text>
</TEI.2>
```

The overall structure of a composite text made up of two unitary texts is:

```
<TEI.2>
<TeiHeader> ... </TeiHeader>
<text>
   <front>
      <!-- front matter of composite text goes here. -->
   </front>
   <group>
      <text>
         <front>
            <!-- front matter of first unitary text, if any -->
         </front>
         <body>
            <!-- body of first unitary text  -->
         </body>
         <back>
            <!-- back matter of first unitary text, if any -->
         </back>
      </text>
      <text>
         <body>
            <!-- body of second unitary text -->
         </body>
      </text>
   </group>
   <back>
      <!-- back matter of composite text, if any -->
   </back>
</text>
</TEI.2>
```

Each of these elements is further described in the following subsections. <text>, <body> and <group> are formally declared as follows:

```
<!-- 7:  Top-level parts of default structure                 -->
<!ELEMENT text          - -  (front?, (body | group), back?)

                                                 +(%m.globincl;)
                                                              >
<!ATTLIST text               %a.global;
                             %a.declaring;                    >
<!ELEMENT body          - O  ((%m.divtop;)*, ( (div+ | div0+ |
                             div1+) | ( (%component)+, (div* |
                             div0* | div1*))), (%m.divbot;)*)  >
<!ATTLIST body               %a.global;
                             %a.declaring;                    >
<!ELEMENT group         - O  ((%m.divtop;)*, (text | group)+,
                             (%m.divbot;)*)                   >
<!ATTLIST group              %a.global;
                             %a.declaring;                    >
<!-- This fragment is used in sec. 7.7                        -->
```

Elements <front> and <back> are declared separately, as further discussed in sections 7.4 ('Front Matter') on p. 201 and 7.6 ('Back Matter') on p. 205. Textual elements, such as paragraphs, lists or phrases, which nest within these major structural elements, are discussed in chapter 6 ('Elements Available in All TEI Documents') on p. 119 (for elements common to all kinds of document) and in part II (for elements specific to a particular base). The <group> element, used for composite texts, is further discussed in section 7.3 ('Groups of Texts') on p. 195.

7.1 Divisions of the Body

In some texts, the body consists simply of a sequence of low-level structural items, referred to here as *components* or *component-level elements* (see section 3.7 ('Element Classes') on p. 51). Examples in prose texts include paragraphs or lists; in dramatic texts, speeches and stage directions; in dictionaries, dictionary entries. In other cases sequences of such elements will be grouped together hierarchically into textual divisions and subdivisions, such as chapters or sections. The names used for these structural subdivisions of texts vary with the genre and period of the text, or even with the whim of the author, editor or publisher. For example, a major subdivision of an epic or of the Bible is generally called a "book", that of a report is usually called a "part" or "section", that of a novel a "chapter" — unless it is an epistolary novel, in which case it may be called a "letter". Even texts which are not organized as linear prose narratives, or not as narratives at all, will frequently be subdivided in a similar way: a drama into "acts" and "scenes"; a reference book into "sections"; a diary or day book into "entries"; a newspaper into "issues" and "sections", and so forth.

To cater for this variety, these Guidelines propose that all such textual divisions be regarded as occurrences of the same neutrally named elements, with an attribute type used to categorize elements independently of their hierarchic level. Two alternative styles are provided for the marking of these neutral divisions: *numbered* and *un-numbered*. Numbered divisions are named <div0>, <div1>, <div2>, etc., where the number indicates the depth of this particular division within the hierarchy, the largest such division being "div0", any subdivision within it being "div1", any further sub-sub-division being "div2" and so on. Un-numbered divisions are simply named , and allowed to nest recursively to indicate their hierarchic depth. The two styles may *not* be combined within a single <front>, <body> or <back> element.

7.1.1 Un-numbered Divisions

The following element is used to identify textual subdivisions in the un-numbered style:

** ** contains a subdivision of the front, body, or back of a text.

As a member of the class *divn*, this element has the following additional attribute:

type specifies a name conventionally used for this level of subdivision, e.g. "act", "volume", "book", "section", "canto", etc.

Using this style, the body of a text containing two parts, each composed of two chapters, might be represented as follows:

```
<body>
<div type='part' n='1'>
      <div type='chapter' n='1'>
      <!-- text of part 1, chapter 1 -->
      </div>
      <div n='2'>
      <!-- text of part 1, chapter 2-->
      </div>
</div>
<div type='part' n='2'>
      <div n='1' type='chapter'>
      <!-- text of part 2, chapter 1 -->
      </div>
      <div n='2'>
      <!-- text of part 2, chapter 2 -->
      </div>
</div>
</body>
```

Note that end-tags are mandatory for un-numbered divisions, to avoid ambiguity. Note also that the type attribute must be specified each time its value changes, for reasons discussed in section 7.1.3 ('Numbered or Un-numbered?') on p. 187 below.

The element has the following formal definition:

```
<!-- 7.1.1:  Un-numbered divisions                                  -->
<!ELEMENT div           - O  ((%m.divtop;)*, (div+ |
                              ((%component;)+, div*)),
                              (%m.divbot;)*)                        >
<!ATTLIST div                %a.global;
                             %a.declaring;
                             %a.divn;                               >
<!-- This fragment is used in sec. 7.7                              -->
```

7.1.2 Numbered Divisions

The following elements are used to identify textual subdivisions in the numbered style:

<div0> contains the largest possible subdivision of the body of a text.

<div1> contains a first-level subdivision of the front, body, or back of a text (the largest, if <div0> is not used, the second largest if it is).

<div2> contains a second-level subdivision of the front, body, or back of a text.

<div3> contains a third-level subdivision of the front, body, or back of a text.

<div4> contains a fourth-level subdivision of the front, body, or back of a text.

<div5> contains a fifth-level subdivision of the front, body, or back of a text.

<div6> contains a sixth-level subdivision of the front, body, or back of a text.

<div7> contains the smallest possible subdivision of the front, body or back of a text, larger than a paragraph.

As members of the class *divn* these elements all bear the following additional attribute:

type specifies a name conventionally used for this level of subdivision, e.g. "act", "volume", "book", "section", "canto", etc.

The largest possible subdivision of the body may be regarded either as a <div0> or as a <div1> element, [1] and the smallest possible <div7>. If numbered divisions are in use, a division at any one level (say, <div3>), may contain only numbered divisions at the next lowest level (in this case, <div4>).

Using this style, the body of a text containing two parts, each composed of two chapters, might be represented as follows:

```
<body>
<div0 type='Part' n='1'>
    <div1 type='Chapter' n='1'>
    <!-- text of part 1, chapter 1 -->
    <div1 n='2'>
    <!-- text of part 1, chapter 2-->
<div0 n='2'>
    <div1 n='1'>
    <!-- text of part 2, chapter 1 -->
    <div1 n='2'>
    <!-- text of part 2, chapter 2 -->
</body>
```

Formal definitions for these elements are as follows:

```
<!-- 7.1.2:  Numbered divisions                                     -->
<!ELEMENT div0          - O  ((%m.divtop;)*, ( div1+ | (
                              (%component;)+, div1*)),
                              (%m.divbot;)*)                        >
<!ATTLIST div0               %a.global;
                             %a.declaring;
                             %a.divn;                               >
<!ELEMENT div1          - O  ((%m.divtop;)*, ( div2+ |
                              ((%component;)+, div2*)),
```

[1] This convention (corresponding with the idea that a type-set document may begin either with a "level 0" or a "level 1" heading) is provided for convenience and compatibility with some widely used formatting systems.

```
                          (%m.divbot;)*)                    >
<!ATTLIST div1            %a.global;
                          %a.declaring;
                          %a.divn;                          >
<!ELEMENT div2       - O  ((%m.divtop;)*, ( div3+ |
                          ((%component;)+, div3*)),
                          (%m.divbot;)*)                    >
<!ATTLIST div2            %a.global;
                          %a.declaring;
                          %a.divn;                          >
<!ELEMENT div3       - O  ((%m.divtop;)*, ( div4+ |
                          ((%component;)+, div4*)),
                          (%m.divbot;)*)                    >
<!ATTLIST div3            %a.global;
                          %a.declaring;
                          %a.divn;                          >
<!ELEMENT div4       - O  ((%m.divtop;)*, ( div5+ |
                          ((%component;)+, div5*)),
                          (%m.divbot;)*)                    >
<!ATTLIST div4            %a.global;
                          %a.declaring;
                          %a.divn;                          >
<!ELEMENT div5       - O  ((%m.divtop;)*, ( div6+ |
                          ((%component;)+, div6*)),
                          (%m.divbot;)*)                    >
<!ATTLIST div5            %a.global;
                          %a.declaring;
                          %a.divn;                          >
<!ELEMENT div6       - O  ((%m.divtop;)*, (div7+ |
                          ((%component;)+, div7*)),
                          (%m.divbot;)*)                    >
<!ATTLIST div6            %a.global;
                          %a.declaring;
                          %a.divn;                          >
<!ELEMENT div7       - O  ((%m.divtop;)*, (%component;)+,
                          (%m.divbot;)*)                    >
<!ATTLIST div7            %a.global;
                          %a.declaring;
                          %a.divn;                          >
<!-- This fragment is used in sec. 7.7                     -->
```

7.1.3 Numbered or Un-numbered?

Within the same <front>, <body> or <back> element, all hierarchic subdivisions must be marked either using nested elements, or using the <div0>, <div1>, <div2> tag appropriate at each level; the two styles may *not* be mixed.

The choice between numbered and un-numbered divisions will depend to some extent on the complexity of the material: un-numbered divisions allow for an arbitrary depth of nesting, while numbered divisions limit the depth of the tree which can be constructed. Where divisions at different levels should be processed differently (chapters, but not sections, for example, beginning on new pages), numbered divisions slightly simplify the task of defining the desired processing for each level. Some software may find numbered divisions easier to process, as there is no need to maintain knowledge of the whole document structure in order to know the level at which a division occurs; such software may however find it difficult to cope with some other aspects of the TEI scheme. On the other hand, in a collection of many works it may prove difficult or impossible to ensure that the same numbered division always corresponds with the same type of textual feature: a "chapter" may be at level 1 in one work and level 3 in another.

Whichever style is used, the global n and id attributes (section 3.5 ('Global Attributes') on p. 42) may be used to provide reference strings or labels for each division of a text, where

appropriate. Such labels should be provided for each section which is regarded as significant for referencing purposes (on reference systems, see further section 6.9 ('Reference Systems') on p. 155).

As indicated above, the type attribute is used to provide a name or description for the division. Typical values might be "book", "chapter", "section", "part", or (for verse texts) "book", "canto", "stanza", or (for dramatic texts) "act", "scene". This attribute has a declared value of #CURRENT, which implies that if defaulted, the value used will be that most recently specified on any element of the same kind, scanning the text left to right. Hence, if un-numbered divisions are used, the appropriate value must be specified each time a change of level occurs, both "down" and "up" the document hierarchy.

The following extended example uses numbered divisions to indicate the structure of a novel, and illustrates the use of the attributes discussed above. It also uses some elements discussed in section 7.2 ('Elements Common to All Divisions') on p. 190 and the <p> element discussed in section 6.1 ('Paragraphs') on p. 120.

```
<div0 type='book' n='I' id=JA0100>
<head>Book I.</head>
  <div1 type='chapter' n='1' id=JA0101>
  <head>Of writing lives in general, and particularly of
        Pamela, with a word by the bye of Colley
        Cibber and others.</head>
  <p>It is a trite but true observation, that examples work
        more forcibly on the mind than precepts:  ...

        <!-- remainder of chapter 1 here -->

  <div1 n='2' id=JA0102>
  <head>Of Mr. Joseph Andrews, his birth, parentage,
        education, and great endowments; with a word
        or two concerning ancestors.</head>
  <p>Mr. Joseph Andrews, the hero of our ensuing history,
        was esteemed to be the only son of Gaffar and
        Gammar Andrews, and brother to the illustrious
        Pamela, whose virtue is at present so famous ...

        <!-- remainder of chapter 2 here -->
        <!-- remaining chapters of Book 1 here -->

<trailer>The end of the first Book
<div0 n='II' id=JA0100>
<head>Book II</head>
  <div1 n='1' id=JA0201>
  <head>Of divisions in authors</head>
  <p>There are certain mysteries or secrets in all
        trades, from the highest to the lowest, from
        that of <term>prime-ministering</term>, to
        this of <term>authoring</term>, which are
        seldom discovered unless to members of
        the same calling ...
  <p>I will dismiss this chapter with the following
        observation: that it becomes an author generally to
        divide a book, as it does a butcher to joint his meat,
        for such assistance is of great help to both the reader
        and the carver. And now having indulged myself a little
        I will endeavour to indulge the curiosity of my reader,
        who is no doubt impatient to know what he will find
        in the subsequent chapters of this book.
  <div1 n='2' id=JA0202>
  <head>A surprising instance of Mr. Adams's short memory, with
        the unfortunate consequences which it brought on Joseph.
```

```
      </head>
<p>Mr. Adams and Joseph were now ready to depart different
      ways ...
```

7.1.4 Partial and Composite Divisions

In most situations, the textual subdivisions marked by elements will be both complete and identically organized with reference to the original source. For some purposes however, in particular where dealing with unusually large or unusually small texts, encoders may find it convenient to present as textual divisions sequences of text which are incomplete with reference to the original text, or which are in fact an ad hoc agglomeration of tiny texts. Moreover, in some kinds of texts it is difficult or impossible to determine the order in which individual subdivisions should be combined to form the next higher level of subdivision, as noted below.

To overcome these problems, the following additional attributes are defined for all elements in the *divn* class:

org specifies how the content of the division is organized. Legal values are:

- ***composite*** composite content: i.e. no claim is made about the sequence in which the immediate contents of this division are to be processed, or their inter-relationships.
- ***uniform*** uniform content: i.e. the immediate contents of this element are regarded as forming a logical unit, to be processed in sequence.

sample indicates whether this division is a sample of the original source and if so, from which part. Legal values are:

- ***initial*** division lacks material present at end in source.
- ***medial*** division lacks material at start and end.
- ***final*** division lacks material at start.
- ***unknown*** position of sampled material within original unknown.
- ***complete*** division is not a sample.

part specifies whether or not the division is fragmented by some other structural element, for example a speech which is divided between two or more verse stanzas. Legal values are:

- ***Y*** the division is incomplete in some respect
- ***N*** either the division is complete, or no claim is made as to its completeness.
- ***I*** the initial part of an incomplete division
- ***M*** a medial part of an incomplete division
- ***F*** the final part of an incomplete division

For example, an encoder might choose to transcribe only the first two thousand words of each chapter from a novel. In such a case, each chapter might conveniently be regarded as a partial division, and tagged with a element in the following form:

```
<div n='xx' sample=Y part=initial>
```

where "xx" represents a number for the chapter. The <sampling> element in the TEI Header should also be used to record the principles underlying the selection of incomplete samples, as further described in section 5.3.2 ('The Sampling Declaration') on p. 95.

The following example demonstrates how a newspaper column composed of very short unrelated snippets may be encoded using these attributes:

```
<div1 type=storylist org=composite>
<head>News in brief</head>
<div2 type=story>
<head>Police deny <soCalled>losing</soCalled> bomb</head>
<p>Scotland Yard yesterday denied claims in the Sunday
Express that anti-terrorist officers trailing an IRA van
loaded with explosives in north London had lost track of
it 10 days ago.
</div2>
<div2 type=story>
```

```
<head>Hotel blaze</head>
<p>Nearly 200 guests were evacuated before dawn
yesterday after fire broke out at the Scandic
Crown hotel in the Royal Mile, Edinburgh.
</div2>
<div2 type=story>
<head>Test match split</head>
<p>Test Match Special next summer will be split
between Radio 5 and Radio 3, after protests this
year that it disrupted Radio 3's music schedule.
</div2>
<!-- other stories here -->
</div1>
```

The org attribute on the <div1> element is used here to indicate that individual stories in this group, marked here as <div2>, are really quite independent of each other, although they are all marked as subdivisions of the whole group. They can be read in any order without affecting the sense of the piece; indeed, in some cases, divisions of this nature are printed in such a way as to make it impossible to determine the order in which they are intended to be read. Individual stories can be added or removed without affecting the existing components.

This method of encoding composite texts as composite divisions has some limitations compared with the more general and powerful mechanisms discussed in section 7.3 ('Groups of Texts') on p. 195. However, it may be preferable in some circumstances, notably where the individual texts are very small.

7.2 Elements Common to All Divisions

The divisions of any kind of text may sometimes begin with a brief heading or descriptive title, with or without a byline, an epigraph or brief quotation, or a salutation such as one finds at the start of a letter. They may also conclude with a brief trailer, byline, or signature. Elements which may appear in this way, either at the start or at the end of a text division proper, are regarded as forming a class, known as *divtop* or *divbot* respectively.

The following special-purpose elements are provided to mark features which may appear only at the start of a division:

<head> contains any heading, for example, the title of a section, or the heading of a list or glossary. Attributes include:

type categorizes the heading in some way meaningful to the encoder.

<epigraph> contains a quotation, anonymous or attributed, appearing at the start of a section or chapter, or on a title page.

<argument> A formal list or prose description of the topics addressed by a subdivision of a text.

<opener> groups together dateline, byline, salutation, and similar phrases appearing as a preliminary group at the start of a division, especially of a letter.

For further details of the <head> element, see section 7.2.1 ('Headings and Trailers') on p. 191; for <epigraph> and <argument>, see section 7.2.3 ('Arguments and Epigraphs') on p. 193; for <opener>, see section 7.2.2 ('Openers and Closers') on p. 192.

The following special-purpose elements are provided to mark features which may appear only at the end of a division:

<trailer> contains a closing title or footer appearing at the end of a division of a text.

<closer> groups together dateline, byline, salutation, and similar phrases appearing as a final group at the end of a division, especially of a letter.

For further details of the <trailer> element, see section 7.2.1 ('Headings and Trailers') on p. 191; for the <closer> element, section 7.2.2 ('Openers and Closers') on p. 192.

7.2.1 Headings and Trailers

The <head> element is used to identify a heading prefixed to the start of any textual division, at any level. A given division may of course contain more than one such element, as in the following example:

```
<div1 n='Etym'>
<head>Etymology</head>
<head>(Supplied by a late consumptive usher to a
grammar school)</head>
<p>The pale Usher &mdash; threadbare in coat, heart,
body and brain; I see him now. He was ever
dusting his old lexicons and grammars....
```

Unlike some other markup schemes, the TEI scheme does *not* require that headings attached to textual subdivisions at different hierarchic levels have different identifiers. All kinds of heading are marked identically using the <head> tag; the type or level of heading intended is implied by the immediate parent of the <head> element, which may for example be a <div1>, <div2>, etc., an un-numbered , or a <list>.

In certain kinds of text (notably newspapers), there may be a need to categorize individual headings within the sequence at the start of a division, for example as "main" headings, or "detail" headings. Specific elements are provided for certain kinds of heading-like features, (notably <byline>, <dateline> and <salute>; see further section 7.2.2 ('Openers and Closers') on p. 192), but the type attribute must be used to discriminate among other forms of heading.

In the following example, taken from a British newspaper, the lead story and its associated headlines have been encoded as a element, with appropriate *divtop* elements attached:

```
<div type=story>
<head rend='large underlined' type=sub>
President pledges safeguards for 2,400 British
troops in Bosnia</head>
<head rend='very large bold' type=main>
Major agrees to enforced no-fly zone </head>
<byline>By George Jones, Political Editor, in Washington
</byline>
<p>Greater Western intervention in the conflict in
former Yugoslavia was pledged by President Bush yesterday....
```

In older writings, the headings or *incipits* may be longer than in modern works. When heading-like material appears in the middle of a text, the encoder must decide whether or not to treat it as the start of a new division. If the phrase in question appears to be more closely connected with what follows than with what precedes it, then it may be regarded as a heading and tagged as the <head> of a new element. If it appears to be simply inserted or superimposed — as for example the kind of "pull quotes" often found in newspapers or magazines, then the <quote>, <q>, or <cit> element may be more appropriate.

The <trailer> element, which can appear at the end of a division only, is used to mark any heading-like feature appearing in this position, as in this example:

```
<div1 type=book n='I'><head>In the name of Christ here begins
the first book of the ecclesiastical history of Georgius
Florentinus, known as Gregory, Bishop of Tours.</head>
<div2><head>Chapter-Headings</head>
<list>
   <!-- list of chapter heads omitted ...  -->
</list>
<div2><head>In the name of Christ here begins Book I of the
history.</head>
```

```
<p>Proposing as I do ...
<p>From the Passion of our Lord until the death of Saint Martin
four hundred and twelve years passed.
<trailer>Here ends the first Book, which covers five thousand,
five hundred and ninety-six years from the beginning of the
world down to the death of Saint Martin.</trailer>
</div2>
</div1>
```

7.2.2 Openers and Closers

In addition to headings of various kinds, divisions sometimes include more or less formulaic opening or closing passages, typically conveying such information as the name and address of the person to whom the division is addressed, the place or time of its production, a salutation or exhortation to the reader, and so on. Divisions in epistolary form are particularly liable to include such features. Additional elements for the detailed encoding of personal names, dates and places are provided in chapter 20 ('Names and Dates') on p. 487. For simple cases, the following elements should be adequate:

<byline> contains the primary statement of responsibility given for a work on its title page or at the head or end of the work.

<dateline> contains a brief description of the place, date, time, etc. of production of a letter, newspaper story, or other work, prefixed or suffixed to it as a kind of heading or trailer.

<salute> contains a salutation or greeting prefixed to a foreword, dedicatory epistle or other division of a text, or the salutation in the closing of a letter, preface, etc.

<signed> contains the closing salutation, etc., appended to a foreword, dedicatory epistle, or other division of a text.

The <byline> and <dateline> elements are used to encode headings which identify the authorship and provenance of a division. Although the terminology derives from newspaper usage, there is no implication that <dateline> or <byline> elements apply only to newspaper texts. The following example illustrates use of the <dateline> and <signed> elements at the end of the preface to a novel:

```
<div type=preface>
<head>To Henry Hope.</head>
<p>It is not because this volume was conceived and partly
executed amid the glades and galleries of the Deepdene,
that I have inscribed it with your name.... I shall find a
reflex to their efforts in your own generous spirit and
enlightened mind.
</p>
<closer>
<signed lang=el>D.</signed>
<dateline>Grosvenor Gate, May-Day, 1844</dateline>
</closer>
</div>
```

Where a sequence of such elements appear together, either at the beginning or end of an element, it may be convenient to group them together using, one of the following elements:

<opener> groups together dateline, byline, salutation, and similar phrases appearing as a preliminary group at the start of a division, especially of a letter.

<closer> groups together dateline, byline, salutation, and similar phrases appearing as a final group at the end of a division, especially of a letter.

The following examples demonstrate the use of the <opener> and <closer> grouping elements:

```
<div type=narrative n='6'>
<head>Sixth Narrative</head>
<head>contributed by Sergeant Cuff</head>
<div type=fragment n='6.1'>
<opener>
  <dateline><name type=place>Dorking, Surrey,</name>
            <date>July 30th, 1849</date>
  </dateline>
  <salute>To <name>Franklin Blake, Esq.</name> Sir, &mdash;
  </salute>
</opener>
<p>I beg to apologize for the delay that has occurred
in the production of the Report, with which I engaged
to furnish you. I have waited to make it a complete
Report ...
<!-- .... -->
<closer>
  <salute>I have the honour to remain, dear sir, your
    obedient servant </salute>
  <signed> <name>RICHARD CUFF</name> (late sergeant in the
     Detective Force, Scotland Yard, London).
  </signed>
</closer>
</div>
```

```
<div type=letter n='14'>
<head>Letter XIV: Miss Clarissa Harlowe to Miss Howe</head>
<opener>
   <dateline>Thursday evening, March 2.</dateline>
</opener>
<p>On Hannah's depositing my long letter ...
<p>An interruption obliges me to conclude myself
in some hurry, as well as fright, what I must ever be,
<closer>
<salute>Yours more than my own,</salute>
<signed>Clarissa Harlowe</signed>
</closer>
</div>
```

For further discussion of the encoding of names of persons and places and of dates, see section 6.4.4 ('Dates and times') on p. 137 and chapter 20 ('Names and Dates') on p. 487.

7.2.3 Arguments and Epigraphs

The <argument> element may be used to encode the prefatory list of topics sometimes found at the start of a chapter or other division. It is most conveniently encoded as a list, since this allows each item to be distinguished, but may also simply be presented as a paragraph. The following are thus both equally valid ways of encoding the same argument:

```
<div type=chap n='6'>
<argument>
<p>Kingston &mdash; Instructive remarks on early English
history &mdash; Instructive observations on carved oak
and life in general &mdash; Sad case of Stivvings,
junior &mdash; Musings on antiquity &mdash; I forget
that I am steering &mdash; Interesting result &mdash;
Hampton Court Maze &mdash; Harris as a guide.
</argument>
<p> It was a glorious morning, late spring or early summer,
as you care to take it...
</div>
```

```
<div type=chap n='6'>
<argument>
<list type='inline'>
<item>Kingston
<item>Instructive remarks on early English history
<item>Instructive observations on carved oak and life in general
<item>Sad case of Stivvings, junior
<item>Musings on antiquity
<item>I forget that I am steering
<item>Interesting result
<item>Hampton Court Maze
<item>Harris as a guide.
</list>
</argument>
<p> It was a glorious morning, late spring or early summer,
as you care to take it...
</div>
```

An *epigraph* is a quotation from some other work appearing on a title page, or at the start of a division. It may be encoded using the special-purpose <epigraph> element. Its content will generally be a <q> or <quote> element, often associated with a bibliographic reference, as in the following example:

```
<div n='19'><head>Chapter 19</head>
 <epigraph>
 <cit><quote>I pity the man who can travel
     from Dan to Beersheba, and say <q>'Tis all
     barren;</q> and so is all the world to him
     who will not cultivate the fruits it offers.
     </quote>
     <bibl>Sterne: Sentimental Journey.</bibl>
 </cit></epigraph>
 <p>To say that Deronda was romantic would be to
 misrepresent him: but under his calm and somewhat
 self-repressed exterior ...
```

For discussion of quotations appearing other than as epigraphs refer to section 6.3.3 ('Quotation') on p. 127.

7.2.4 Content of Textual Divisions

Other than its initial sequence of *divtop* elements, and its closing sequence of *divbot* elements, every textual division (numbered or un-numbered) consists of a sequence of ungrouped *component* elements (see 3.7 ('Element Classes') on p. 51). The actual elements available will depend on the base tag set in use; in all cases, at least the component-level structural elements defined in the core will be available (paragraphs, lists, dramatic speeches, verse lines and line groups etc.). If the drama base has been selected, then additionally the low level dramatic structural elements (speeches or stage directions, as defined in chapter 10 ('Base Tag Set for Drama') on p. 227) will be available. If the dictionary base is in use, then dictionary entries, related entries, etc. (as defined in chapter 12 ('Print Dictionaries') on p. 269) will also be available; if the tag set for transcribed speech is in use, then utterances, pauses, vocals, kinesics, etc., as defined in chapter 11.2 ('Elements Unique to Spoken Texts') on p. 253 will be available; and so on.

Where a text contains low level elements from more than one base, two options are available. The first option, selected by the "mixed" base, allows for low level structural elements from any or all of the selected bases to appear at any point. The second option, selected by the "general" base, allows for low level structural elements from different bases to appear in different textual divisions of the same text, but requires that any one division use elements from only one base. For further information, refer to section 3.4 ('Combining TEI Base Tag Sets') on p. 40.

The elements discussed in this section are formally defined as follows:

```
<!-- 7.2.4:  Tags for start and end of divisions              -->
<!ELEMENT trailer       - O  (%phrase.seq;)                    >
```

```
<!ATTLIST trailer            %a.global;                          >
<!ELEMENT byline        - O  (%phrase.seq; | docAuthor)*         >
<!ATTLIST byline             %a.global;                          >
<!ELEMENT dateline      - O  (date | time | name | #PCDATA |
                             address)*                           >
<!ATTLIST dateline           %a.global;                          >
<!ELEMENT argument      - -  (head?, %component.seq;)            >
<!ATTLIST argument           %a.global;                          >
<!ELEMENT epigraph      - -  (%component.seq;)                   >
<!ATTLIST epigraph           %a.global;                          >
<!ELEMENT opener        - O  (signed | dateline | salute |
                             %phrase.seq;)*                      >
<!ATTLIST opener             %a.global;                          >
<!ELEMENT closer        - O  (signed | dateline | salute |
                             %phrase.seq;)*                      >
<!ATTLIST closer             %a.global;                          >
<!ELEMENT salute        - O  (%phrase.seq;)                      >
<!ATTLIST salute             %a.global;                          >
<!ELEMENT signed        - O  (%phrase.seq;)                      >
<!ATTLIST signed             %a.global;                          >
<!-- The HEAD element is declared in the core tag set.        -->

<!-- This fragment is used in sec. 7.7                        -->
```

7.3 Groups of Texts

The <group> element should be used to represent a collection of independent texts which is to be regarded as a single unit for processing or other purposes. Examples of such composite texts include anthologies and other collections. The presence of common front matter referring to the whole collection, possibly in addition to front matter relating to each individual text, is a good indication that a given text might usefully be encoded as a <group>, though encoders may choose to use this structure to represent other kinds of composite texts as well.

<group> contains the body of a composite text, grouping together a sequence of distinct texts (or groups of such texts) which are regarded as a unit for some purpose, for example the collected works of an author, a sequence of prose essays, etc.

For example, the overall structure of a collection of short stories might be encoded as follows:

```
<TEI.2>
<TEIHeader>
   <!-- header information for the whole collection -->
</TEIHeader>
<text>
   <front>
      <docTitle>The Adventures of Sherlock Holmes
      </docTitle>
      <note>First published in <title>The Strand</title>
      between July 1891 and December 1892</note>
      <!-- Any other front matter specific
           to the collection here ... -->
   </front>
   <group>
      <text>
         <front>
            <head rend=italic>Adventures of Sherlock
                  Holmes</head>
            <docTitle>Adventure I. &mdash;
                  A Scandal in Bohemia</docTitle>
            <byline>By A. Conan Doyle.</byline>
```

```
            </front>
            <body>
               <div1 n='I.1'>
               <p>To Sherlock Holmes she is always
               <emph>the</emph> woman.
               ...
            </body>
         </text>
         <text>
            <front>
               <head rend=italic>Adventures
                     of Sherlock Holmes</head>
               <docTitle>Adventure II. &mdash;
                     The Red-Headed League</docTitle>
               <byline>By A. Conan Doyle.</byline>
            </front>
            <body>
               <!-- text of The Red-Headed League here -->
            </body>
         </text>

         <!-- more texts here -->

         <text>
            <front>
               <head rend=italic>
                   Adventures of Sherlock Holmes</head>
               <docTitle>Adventure XII. &mdash;
                   The Adventure of the Copper Beeches
               </docTitle>
               <byline>By A. Conan Doyle.</byline>
            </front>
            <body>
               <p><q>&odq;To the man who loves art for
               its own sake,&cdq;</q> remarked Sherlock
               Holmes ...
               <!-- rest of the the Copper Beeches here -->
               ... she is now the head of a private school
               at Walsall, where I believe that she has
               met with considerable success.</p>
            </body>
         </text> <!-- end of the Copper Beeches -->
      </group>
   </text>  <!-- end of the Adventures of Sherlock Holmes -->
   </TEI.2>
```

A text which is a member of a group may itself contain groups. This is quite common in collections of verse, but may happen in any kind of text. As an example, consider the overall structure of a typical collection, such as the *Muses Library* edition of Crashaw's poetry (ed. J.R. Tutin, [ca. 1900]). Following a critical introduction and table of contents, this work contains the following major sections:

- *Steps to the Temple* (a collection of verse first published in 1648)
- *Carmen deo Nostro* (a second collection, published in 1652)
- *The Delights of the Muses* (a third collection, published in 1648)
- *Posthumous Poems,* I (a collection of fragments all taken from a single manuscript)
- *Posthumous Poems,* II (a further collection of fragments, taken from a different manuscript)

Each of the three collections published in Crashaw's lifetime has a reasonable claim to be considered as a text in its own right, and may therefore be encoded as such. It is rather more arbitrary as to whether the two posthumous collections should be treated as two groups, following

the practice of the Muses Library edition. An encoder might elect to combine the two into a single group, or simply to treat each fragment as an ungrouped unitary text.

The Muses Library edition reprints the whole of each of the three original collections, including their original front matter (title pages, dedications etc.). These should be encoded using the <front> element and its constituents (on which see further section 7.4 ('Front Matter') on p. 201), while the body of each collection should be encoded as a single <group> element. Each individual poem within the collections should be encoded as a distinct <text> element. The beginning of the whole collection would thus appear as follows (for further discussion of the use of the elements and <lg> for textual subdivision of verse, see section 6.11.1 ('Core Tags for Verse') on p. 177 and chapter 9 ('Base Tag Set for Verse ') on p. 213):

```
<text>
  <front>
    <titlePage>
      <docTitle>The poems of Richard Crashaw</docTitle>
      <byline>Edited by J.R. Tutin</byline>
      ...
    </titlePage>
    <div type=preface><head>Editor's Note
      <p>A few words are necessary...
      ...
  </front>

  <group>
  <text>
    <front>
      <titlepage>
        <docTitle>
          Steps to the Temple, Sacred Poems
        </docTitle>
        ...
      </titlepage>
      <div type=address><head>The Preface to the Reader
        <p>Learned Reader, The Author's friend will not usurp
        much upon thy eye...
      </div>
    </front>
    <group>
      <text>
        <front><docTitle>Sospetto D'Herode</docTitle>
        </front>
        <body>
          <div1 type=book n='Herod I'>
          <head>Libro Primo</head>
          <epigraph>
            <l>Casting the times with their strong signs
            ...
          </epigraph>
          <lg n='I.1' type=stanza>
          <l>Muse! now the servant of soft loves no more
          <l>Hate is thy theme and Herod whose unblest
          <l>Hand (O, what dares not jealous greatness?) tore
          <l>A thousand sweet babes from their mothers' breast,
          <l>The blooms of martyrdom...
          ...
      </text>
      <text>
        <front><docTitle>The Tear</docTitle></front>
        <body>
```

```
        <lg n='I'>
        <l>What bright soft thing is this
        <l>Sweet Mary, thy fair eyes' expense?
        ...
      </text>
```

Following the remaining poems of the *Steps to the Temple*, each one within its own <text> element, the structure of the remainder of the work might be represented as follows:

```
    </group>
    <back> <!-- back matter for Steps to the Temple here -->
    </back>
  </text>
  <text> <!-- Carmen deo Nostro -->
    <front> ... </front>
    <group>
      <text>... </text>
      <text>... </text>
      <!-- more texts here -->
    </group>
  </text>
  <text> <!-- The delights of the Muses -->
    <group>
      <text>...  </text>
      <text>...  </text>
      <!-- more texts here -->
  </text>
   ...
  </group>
  <back>
    <!-- back matter for the whole collection -->
  </back>
</text>
```

The <group> element may be used in this way to encode any kind of collection of which the constituents are regarded by the encoder as texts in their own right. Examples include anthologies of verse or prose by multiple authors, collections, florilegia or commonplace books, journals, day books, etc. As a fairly typical example, we consider *The Norton Book of Travel*, an anthology edited by Paul Fussell and published in 1987 by W.W. Norton. This work comprises the following major sections:

1. Front matter (title page, acknowledgments, introductory essay)
2. The Beginnings
3. The Eighteenth Century and the Grand Tour
4. The Heyday
5. Touristic Tendencies
6. Post Tourism
7. Back matter (permissions list, index)

Each titled section listed above comprises a group of extracts or complete texts from writers of a given historical period, preceded by an introductory essay. For example, the second group listed above contains, inter alia, the following:

1. Prefatory essay
2. Five letters by Lady Mary Wortley Montagu
3. An extract from Swift's Gullivers Travels
4. Two poems by Alexander Pope
5. Two extracts from Boswell's Journal
6. A poem by William Blake

Each group of writings by a single author is preceded by a brief biographical notice. Some of the extracts are quite lengthy, containing several chapters or other divisions; others are quite

short. As the above list indicates, the texts included range across all kinds of material: verse, prose, journals and letters.

The easiest way of encoding such an anthology is to treat each individual extract as a text in its own right. A sequence of texts by a single author, together with the biographical note preceding it, can then be treated as a single <group> element within the larger <group> formed by the section. The sequence of single or composite texts making up a single section of the work is likewise treated, together with its prefatory essay, as a single <group> within the work. Schematically:

```
<text> <!-- the whole anthology -->
  <front>
      <!-- title page, acknowledgments, introductory
           essay for anthology -->
  </front>
  <group> <!-- 'body' of the anthology -->
    <group><head>The Beginnings</head>
          <!-- sequence of texts or groups -->
    </group>
    <group> <!-- The Eighteenth Century and the Grand Tour -->
       <text> <!-- prefatory essay by editor -->  </text>
       <group><!-- Lady Mary Wortley Montagu -->
          <text> <!-- biographical notice, by
                      editor --> </text>
          <text> <!-- first letter  --> </text>
          <text> <!-- second letter --> </text>
           ...
       </group>  <!-- end of Montagu section -->
       <text>    <!-- single text by Jonathan Swift -->
            <front> <!-- biographical notice,
                         by editor --> </front>
            <body> ... </body>
       </text>  <!-- end of Swift section -->
       <group>  <!-- Alexander Pope -->
          <text> <!-- biographical notice, by
                      editor --> </text>
          <text> <!-- first poem  --> </text>
          <text> <!-- second poem --> </text>
       </group> <!-- end of Pope section -->
       ...
    </group> <!-- end of 18th Century Section -->
    <group><head>The Heyday</head>
        <!-- texts and subgroups ... -->
    </group>
    ...
  </group> <!-- end of 'body' of anthology -->
  <back> <!-- back matter for whole anthology  -->
  </back>
</text>  <!-- end of the anthology -->
```

Note that the editor's introductory essays on each author may be treated as texts in their own right, (as the essays on Lady Mary Wortley Montagu and Alexander Pope have been treated above), or as front matter to the embedded text, as the essay on Swift has been. The treatment in the example is intentionally inconsistent, to allow comparison of the two approaches. Consistency can be imposed either by treating the Swift section as a <group> containing one text by Swift and one by the editor, or by treating the Montagu and Pope sections as <text> elements containing the editor's essays as front matter. Marked in the second way, the Pope section of the book would look like this:

```
    <text><!-- Alexander Pope -->
        <front>
            <!-- biographical notice -->
        </front>
```

```
        <group>
             <text>  <!-- first poem  --> </text>
             <text>  <!-- second poem --> </text>
             ...
        </group>
    </text> <!-- end of Pope section -->
```

The essays on "The Eighteenth Century and the Grand Tour" and other larger sections could also be tagged as "front" matter in the same way, by treating the larger sections as **<text>** elements rather than **<group>** elements.

Where, as in this case, an anthology contains different kinds of text (for example, mixtures of prose and drama, or transcribed speech and dictionary entries, or letters and verse), the elements to be encoded may well need to be drawn from more than one of the base tag sets described in part II. In such a situation, either the mixed or the general base should be specified, as further described in section 3.4 ('Combining TEI Base Tag Sets') on p. 40. The elements provided by the core tag set described in chapter 6 ('Elements Available in All TEI Documents') on p. 119 should however prove adequate for most simple purposes, where prose, drama, and verse are combined in a single collection.

For anthologies of short extracts such as commonplace books, it may often be preferable to regard each extract not as a text in its own right but simply as a quotation or **<cit>** element. The following component-level elements may be used to encode quotations of this kind:

<cit> A quotation from some other document, together with a bibliographic reference to its source.

<quote> contains a phrase or passage attributed by the narrator or author to some agency external to the text.

For example, the chapter of "extracts" which appears in the front matter of Melville's *Moby Dick* might be encoded as follows:

```
<div type=chap n='2'>
<head>Extracts</head>
<head>(Supplied by a sub-sub-Librarian)</head>

<p>It will be seen that this mere painstaking burrower and
grubworm of a poor devil of a Sub-Sub appears to have gone
through the long Vaticans and street-stalls of the earth,
picking up whatever random allusions to whales he could
anyways find...
Here ye strike but splintered hearts together &dash; there,
ye shall strike unsplinterable glasses!</p>

<cit>
  <quote>And God created great whales.</quote>
  <bibl>Genesis</bibl>
</cit>
<cit>
  <quote>
    <l>Leviathan maketh a path to shine after him;
    <l>One would think the deep to be hoary.
  </quote>
  <bibl>Job</bibl>
  ...
<cit>
  <quote>By art is created that great Leviathan,
  called a Commonwealth or State &dash (in Latin,
  <mentioned lang=LAT>civitas</mentioned>), which
  is but an artificial man.
  </quote>
  <bibl>Opening sentence of Hobbes's Leviathan</bibl>
</cit>
```

For more information on the use of the <quote> and <bibl> elements, see sections 6.3.3 ('Quotation') on p. 127 and 6.10 ('Bibliographic Citations and References') on p. 162 respectively.

Where one or more whole texts are embedded within other texts, without necessarily forming a composite, the encoder may also choose to represent the nested structure directly. The <text> element is itself a *component* element, and thus can appear within any division level element in the same way as a paragraph. For example, texts such as the *Decameron* or the *Arabian Nights* might be regarded as sequences of discrete texts embedded within another single text, the framing narrative, rather than as groups of discrete texts in which the fragments of framing narrative are regarded as front matter.

7.4 Front Matter

By *front matter* we mean distinct sections of a text (usually, but not exclusively, a printed one), prefixed to it by way of introduction or identification as a part of its production. Features such as title pages or prefaces are clear examples; a less definite case might be the prologue attached to a play. The front matter of an encoded text should not be confused with the TEI header described in chapter 5 ('The TEI Header') on p. 77, which serves as a kind of front matter for the computer file itself, not the text it encodes.

An encoder may choose simply to ignore the front matter in a text, if the original presentation of the work is of no interest, or for other reasons; alternatively some or all components of the front matter may be thought worth including with the text as components of the <front> element. [2] With the exception of the title page, (on which see section 7.5 ('Title Pages') on p. 203), front matter should be encoded using the same elements as the rest of a text. As with the divisions of the text body, no other specific tags are proposed here for the various kinds of subdivision which may appear within front matter: instead either numbered or un-numbered elements may be used. The following suggested values [3] for the type attribute may be used to distinguish various kinds of division characteristic of front matter:

preface A foreword or preface addressed to the reader in which the author or publisher explains the content, purpose or origin of the text.

ack A formal declaration of acknowledgment by the author in which persons and institutions are thanked for their part in the creation of a text.

dedication A formal offering or dedication of a text to one or more persons or institutions by the author.

abstract A summary of the content of a text as continuous prose.

contents A table of contents, specifying the structure of a work and listing its constituents. The <list> element should be used to mark its structure.

frontispiece A pictorial frontispiece, possibly including some text.

The following extended example demonstrates how various parts of the front matter of a text may be encoded. The front part begins with a title page, which is presented in section 7.5 ('Title Pages') on p. 203 below. This is followed by a dedication and a preface, each of which is encoded as a distinct :

```
<div type='dedication'>
<p>To my parents, Ida and Max Fish
</div>

<div type='preface'>
<head>Preface</head>
<p>The answer this book gives to its title question is
<q>there is and there isn't</q>.
```

[2] This decision should be recorded in the <sampling> element of the header.

[3] As with all lists of "suggested values" for attributes, it is recommended that software written to handle TEI-conformant texts be prepared to recognize and handle these values when they occur, without limiting the user to the values in this list.

```
    ...
<p>Chapters 1-12 have been previously published in the
following journals and collections:
<list>
<item>chapters 1 and 3 in <title>New literary
History</title></item>
...
<item>chapter 10 in <title>Boundary II</title> (1980)</item>
</list>. I am grateful for permission to reprint.
<signed>S.F.</signed>
</div>
```

The front matter concludes with another element, shown in the next example, this time containing a table of contents, which contains a <list> element (as described in section 6.7 ('Lists') on p. 149). Note the use of the <ptr> element to provide page-references: the implication here is that the target identifiers supplied (P1, P68 etc.) may correspond with identifiers used either for elements representing chapters of the text, or for <pb> elements marking page divisions of the text. (For the <ptr> element, see 6.6 ('Simple Links and Cross References') on p. 147.) Alternatively, the literal page numbers present in the source text might be transcribed, but they are likely to be of little direct use in work with the electronic text.

```
<div type='contents'>
<head>Contents</head>
<list>
<item>Introduction, or How I stopped Worrying and Learned
  to Love Interpretation <ptr target=p1>
</item>
<item><list><head>Part One: Literature in the Reader</head>
  <item n='1'>Literature in the Reader: Affective
  Stylistics <ptr target=p21>
  <item n='2'>What is Stylistics and Why Are They
  Saying Such Terrible Things About It? <ptr target=p68>
  ...
</list></list>
</div>
```

The following example uses numbered divisions to mark up the front matter of a medieval text. (Entity references are used to represent the characters thorn, yogh, and ampersand, as discussed in section 4.1.2 ('Characters Not Available Locally') on p. 72.) Note that in this case no title page in the modern sense occurs; the title is simply given as a heading at the start of the front matter. Note also the use of the type attribute on the elements to indicate document elements comparatively unusual in modern books such as the initial prayer:

```
<front>
<div1 type=incipit>
<p>Here bygynni&th; a book of contemplacyon, &th;e whiche
is clepyd  <title>&Th;E CLOWDE OF VNKNOWYNG</title>,
in &th;e whiche a soule is onyd wi&th; GOD.
<div1 type='prayer'>
<head>Here biginne&th; &th;e preyer on &th;e prologe.</head>
<p>God, unto whom alle hertes ben open, &amp; unto
whome alle wille speki&th;, &amp; unto whom no priue
&th;ing is hid: I beseche &th;ee so for to clense &th;e
entent of myn hert wi&th; &th;e unspekable &yog;ift of
&th;i grace, &th;at I may parfiteliche loue &th;ee
&amp; wor&th;ilich  preise &th;ee. Amen.

<div1 type='preface'>
<head>Here biginne&th; &th;e prolog.</head>
<p>In &th;e name of &th;e Fader &amp; of &th;e Sone &amp;
of &th;e Holy Goost.
<p>I charge &th;ee &amp; I beseeche &th;ee, wi&th; as moche
```

```
power &amp; vertewe as &th;e bonde of charite is sufficient
to suffre, what-so-euer &th;ou be &th;at &th;is book schalt
haue in possession ...

<div1 type='contents'>
<head>Here biginne&th; a table of &th;e chapitres.</head>
<list>
<label>&th;e first chapitre </label>
   <item>Of foure degrees of Cristen mens leuing;
    &amp; of &th;e cours of his cleping &th;at &th;is
    book was maad vnto.</item>
<label>&th;e secound chapitre</label>
   <item>A schort stering to meeknes &amp; to &th;e
     werk of &th;this book</item>
    <!--  ... -->
<label>&th;e fiue and seuenti chapitre</label>
   <item>Of somme certein tokenes bi &th;e whiche
     a man may proue whe&th;er he be clepid of God
     to worche in &th;is werk.</item>
</list>
<trailer>&amp here eende&th; &th;e table of &th;e chapitres.
</trailer>
</front>
```

7.5 Title Pages

Detailed analysis of the title page and other *preliminaries* of older printed books and manuscripts is of major importance in descriptive bibliography and the cataloguing of printed books; such analysis may require a rather more detailed tag set than that proposed here. [4] The following elements are therefore proposed as an interim measure; they constitute a useful descriptive tag set for the major features of most title pages:

<titlePage> contains the title page of a text, appearing within the front or back matter.

<docTitle> contains the title of a document, including all its constituents, as given on a title page.

<titlePart> contains a subsection or division of the title of a work, as indicated on a title page. Attributes include:

type specifies the role of this subdivision of the title. Suggested values include:

main main title of the work

sub subtitle of the work

alt alternative title of the work

desc descriptive paraphrase of the work included in title

<byline> contains the primary statement of responsibility given for a work on its title page or at the head or end of the work.

<docAuthor> contains the name of the author of the document, as given on the title page (often but not always contained in a <byline>).

<epigraph> contains a quotation, anonymous or attributed, appearing at the start of a section or chapter, or on a title page.

<imprimatur> contains a formal statement authorizing the publication of a work, sometimes required to appear on a title page or its verso.

<docEdition> contains an edition statement as presented on a title page of a document.

[4] Definition of such a tag set remains a work item for the TEI; such tag sets for contemporary printed matter already exist or are being created within the publishing industry, for example the Majour (Modular Application for Journals) Project of the European Workgroup on SGML. See for example *MAJOUR: Modular Application for Journals: DTD for Article Headers* ([n.p.]: EWS, 1991).

<docImprint> contains the imprint statement (place and date of publication, publisher name), as given (usually) at the foot of a title page.

<docDate> contains the date of a document, as given (usually) on a title page.

These elements constitute the element class *tpParts*, which is defined by the parameter entity *m.tpParts*. Encoders wishing to add new elements to this class may do so by modifying or redefining this parameter entity, as further described in chapter 29 ('Modifying the TEI DTD') on p. 619. Two examples of the use of these elements follow. First, the title page of the work discussed earlier in this section:

```
<front>
<titlePage>
<docTitle>
<titlePart type=main>Is There a Text in This Class?</>
<titlePart type=sub>The Authority of Interpretive Communities</>
</docTitle>
<docAuthor>Stanley Fish</>
<docImprint><publisher>Harvard University Press</>
  <pubPlace>Cambridge, Massachusetts</>
  <pubPlace>London, England</>
</docImprint>
</titlepage>
```

Second, a characteristically verbose 17th century example:

```
<titlepage>
<docTitle>
 <titlepart type=main>  THE
  Pilgrim's Progress
  FROM
  THIS WORLD,
  TO
  That which is to come:</>
<titlepart type=sub>
  Delivered under the Similitude of a
  DREAM</>
<titlepart type=desc>  Wherein is Discovered,
  The manner of his setting out,
  His Dangerous Journey; And safe
  Arrival at the Desired Countrey.</>
</docTitle>
<epigraph>
  <cit><q>I have used Similitudes,</q><bibl>Hos. 12.10</bibl></cit>
</epigraph>
<byline>
  By <docAuthor>John Bunyan</>.</byline>
<imprimatur>Licensed and Entred according to Order.</>
<docImprint>
  <pubPlace>LONDON,</>
  Printed for <name>Nath. Ponder</>
  at the <name>Peacock</> in the <name>Poultrey</>
  near <name>Cornhil</>, <docDate>1678</>.
</docImprint>
```

Those elements in the above list which are not defined elsewhere have the following formal declarations:

```
<!-- 7.5:  Tags for title pages                                  -->
<!ELEMENT titlePage     - o  (%m.tpParts;)+                       >
<!ATTLIST titlePage          %a.global;
          type               CDATA               #IMPLIED       >
<!ELEMENT docTitle      - o  (titlePart+)                         >
<!ATTLIST docTitle           %a.global;                           >
<!ELEMENT titlePart     - O  (%paraContent;)                      >
```

```
<!ATTLIST titlePart          %a.global;
          type               CDATA               main          >
<!ELEMENT docAuthor      - O  (%phrase.seq;)                     >
<!ATTLIST docAuthor          %a.global;                          >
<!ELEMENT imprimatur     - O  (%paraContent;)                    >
<!ATTLIST imprimatur         %a.global;                          >
<!ELEMENT docEdition     - O  (%paraContent;)                    >
<!ATTLIST docEdition         %a.global;                          >
<!ELEMENT docImprint     - O  (%phrase.seq | pubPlace | docDate
                              | publisher)*                      >
<!ATTLIST docImprint         %a.global;                          >
<!ELEMENT docDate        - O  (%phrase.seq;)                     >
<!ATTLIST docDate            %a.global;
          value              %ISO-date           #IMPLIED      >
<!-- This fragment is used in sec. 7.5                          -->
```

Where title pages are encoded, their physical rendition is often of considerable importance. One approach to this requirement would be to use the <seg> tag, described in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331, to segment the typographic content of each part of the title page, and then use the global rend attribute to specify its rendition. Another would be to use a tag set specialized for the description of typographic entities such as pages, lines, rules, etc., bearing special-purpose attributes to describe line height, leading, degree of kerning, font, etc. Further discussion of these problems is provided in chapter 18 ('Transcription of Primary Sources') on p. 443.

Front matter elements are defined in a distinct DTD file called *TEIfron2.dtd*.

```
<!-- 7.5:  Additional Tag Set for Front Matter                  -->
<!ELEMENT front          - O  ( (%m.front;)*, ( ( (%m.divtop;),
                              (%m.divtop; | titlePage)*) | (
                              (div), (div | (%m.front;) )*) | (
                              (div1), (div1 | (%m.front;) )*) )?
                              )                                   >
<!ATTLIST front              %a.global;
                              %a.declaring;                       >
<!ENTITY % x.tpParts ''                                          >
<!ENTITY % m.tpParts '%x.tpParts byline | docAuthor | docDate |
           docEdition | docImprint | docTitle | epigraph |
           imprimatur | titlePart'                               >
<!-- ... declarations from section 7.5                          -->
<!--     (Tags for title pages)                                  -->
<!--     go here ...                                             -->
```

7.6 Back Matter

Conventions vary as to which elements are grouped as back matter and which as front. For example, some books place the table of contents at the front, and others at the back. Even title pages may appear at the back of a book as well as at the front. The content model for <back> and <front> elements are therefore identical.

The following suggested values may be used for the type attribute on all division elements, in order to distinguish various kinds of division characteristic of back matter:

appendix An ancillary self-contained section of a work, often providing additional but in some sense extra-canonical text.

glossary A list of terms associated with definition texts ("glosses"): this should be encoded as a <list type=gloss> (see section 6.7 ('Lists') on p. 149).

notes A section in which textual or other kinds of notes are gathered together.

bibliogr A list of bibliographic citations: this should be encoded as a <listBibl> (see section 6.10 ('Bibliographic Citations and References') on p. 162).

index Any form of index to the work.

colophon A statement appearing at the end of a book describing the conditions of its physical production.

No additional elements are proposed for the encoding of back matter at present. Some characteristic examples follow; first, an index (for the case in which a printed index is of sufficient interest to merit transcription):

```
<back>
  <div type=index>
  <head>Index</head>
  <list type=index>
  <item>Actors, public, paid for the contempt attending
    their profession, <ptr target=p209>
  <item>Africa, cause assigned for the barbarous state of
    the interior parts of that continent, <ptr target=p125>
  <item>Agriculture
    <list type=indexentry>
    <item>ancient policy of Europe unfavourable to, <ptr target=p371>
    <item>artificers necessary to carry it on, <ptr target=p481>
    <item>cattle and tillage mutually improve each other,
         <ptr target=p325>
    ...
    <item>wealth arising from more solid than that which proceeds
    from commerce <ptr target=p520>
    </list>
  <item>Alehouses, not the efficient cause of drunkenness,
       <ptr target=p461>
  ...
  </list>
  </div>
  ...
</back>
```

Next, a back-matter division in epistolary form:

```
<back>
  <div type=letter>
  <head>A letter written to his wife, founde with this booke
  after his death.</head>
  <p>The remembrance of the many wrongs offred thee, and thy
  unreproued vertues, adde greater sorrow to my miserable state,
  than I can utter or thou conceiue....
  ... yet trust I in the world to come to find mercie, by the
  merites of my Saiuour to whom I commend thee, and commit
  my soule.
  <signed>Thy repentant husband for his disloyaltie,
  <name>Robert Greene.</name></signed>
  <epigraph lang=LA><p>Faelicem fuisse infaustum</epigraph>
  <trailer>FINIS</trailer>
  </div>
  ...
</back>
```

And finally, a list of corrigenda and addenda with pseudo-epistolary features:

```
  <back>
  <div type=corrigenda>
  <head>Addenda</head>
  <salute lang=LA>M. Scriblerus Lectori</salute>
  <p>Once more, gentle reader I appeal unto thee, from the
  shameful ignorance of the Editor, by whom Our own Specimen
  of <name>Virgil</> hath been mangled in such miserable
  manner, that scarce without tears can we behold it. At the
  very entrance, Instead of <q lang=GR>prolego/mena</q>,
```

```
lo! <q lang=GR>prolegw/mena</q> with an Omega!
and in the same line <q lang=LA>consul&acirc;s</q>
with a circumflex! In the next page thou findest <q lang=LA>
leviter perlabere</>, which his ignorance took to be the
infinitive mood of <q lang=LA>perlabor</> but ought to
be <q lang=LA>perlabi</> ... Wipe away all these monsters,
Reader, with thy quill.</p>
</div>
</back>
```

The <back> element is defined in file *TEIback2.dtd*; since there are no other specialized back-matter tags, nothing else is defined there.

```
<!-- 7.6:  Tags for Back Matter                                  -->
<!ELEMENT back          - O  ( (%m.front)*, ( ( (%m.divtop),
                             (%m.divtop | titlePage)*) | (
                             (div), (div | (%m.front))*) | (
                             (div1), (div1 | (%m.front))*) )? )
                                                                  >
<!ATTLIST back               %a.global;
                             %a.declaring;                        >
```

7.7 DTD Fragment for Default Text Structure

The DTD fragment described by the present chapter is found in file *teistr2.dtd*; it has the following overall structure:

```
<!-- 7.7:  Default text structure                                -->
<!-- This definition of the basic text structure is used by     -->
<!-- most TEI base tag sets; some bases, however, use slight    -->
<!-- variations upon it.                                        -->


<!ENTITY % x.divtop ''                                            >
<!ENTITY % m.divtop '%x.divtop argument | byline | docAuthor |
           docDate | epigraph | head | opener | salute |
           signed'                                                >
<!ENTITY % x.divbot ''                                            >
<!ENTITY % m.divbot '%x.divbot byline | closer | epigraph |
           salute | signed | trailer'                            >

<!-- ... declarations from section 7                            -->
<!--     (Top-level parts of default structure)                 -->
<!--     go here ...                                            -->
<!-- ... declarations from section 7.1.1                        -->
<!--     (Un-numbered divisions)                                -->
<!--     go here ...                                            -->
<!-- ... declarations from section 7.1.2                        -->
<!--     (Numbered divisions)                                   -->
<!--     go here ...                                            -->
<!-- ... declarations from section 7.2.4                        -->
<!--     (Tags for start and end of divisions)                  -->
<!--     go here ...                                            -->

<!-- Front matter is defined in TEI.front file.                 -->

<!ENTITY % TEI.front.dtd system 'teifron2.dtd'                    >
%TEI.front.dtd;

<!-- Back matter is defined in TEI.back file.                   -->
```

```
<!ENTITY % TEI.back.dtd system 'teiback2.dtd'                    >
%TEI.back.dtd;
```

Part III

Base Tag Sets

Chapter 8

Base Tag Set for Prose

This chapter describes the base tag set for prose texts, the simplest of the base tag sets defined by these Guidelines. It should be used for all texts in prose, whether fiction or non-fiction, novel or technical manual.

Prose material can appear in the front and back matter of virtually any text type, even where the body may be composed of very different component elements. Consequently, the elements needed to encode conventional prose are all included in the core tag set described in chapter 6 (‘Elements Available in All TEI Documents’) on p. 119) and the base tag set for prose itself defines no elements at all: it merely invokes the tag set for default text structure. When this base tag set is used, all the elements defined in the core tag sets are available, as are those in the default text structure described in chapter 7 (‘Default Text Structure’) on p. 183. No other elements are available, unless one or more of the additional tag sets is also enabled.

The base tag set for prose should be invoked by defining the parameter entity *tei.prose* with the value “INCLUDE”. A document using this base tag set and no other will thus being as follows:

```
<!DOCTYPE tei.2 system 'tei2.dtd' [
   <!ENTITY % tei.prose 'INCLUDE' >
]>
```

The base tag set for prose is formally defined by the following DTD fragment:

```
<!-- 8:  Base Tag Set for Prose                                  -->
<!-- Text Encoding Initiative: Guidelines for Electronic        -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.      -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy      -->
<!-- in any form is granted, provided this notice is            -->
<!-- included in all copies.                                    -->

<!-- These materials may not be altered; modifications to       -->
<!-- these DTDs should be performed as specified in the         -->
<!-- Guidelines in chapter "Modifying the TEI DTD."             -->

<!-- These materials subject to revision. Current versions      -->
<!-- are available from the Text Encoding Initiative.           -->

<!-- This base tag set does nothing but declare and embed       -->
<!-- the default text structure. When it is selected,           -->
<!-- therefore, the only tags available are those in the        -->
<!-- core and those defined in other selected tag sets.         -->


<![ %TEI.singleBase [
<!ENTITY % TEI.structure.dtd system 'teistr2.dtd'                >
%TEI.structure.dtd;
]]&nil;>
```

Chapter 9

Base Tag Set for Verse

This base tag set is intended for use when encoding texts which are entirely or predominantly in verse, and for which the elements for encoding verse structure already provided by the core tag set are inadequate.

The tags described in section 6.11.1 ('Core Tags for Verse') on p. 177 include elements for the encoding of verse lines and line groups such as stanzas: these are available for any TEI document, irrespective of the base tag set it uses. Like the base tag sets for prose and for drama, the base tag set for verse additionally makes use of the tag set defined in chapter 7 ('Default Text Structure') on p. 183 to define the basic formal structure of a text, in terms of <front>, <body> and <back> elements and the text-division elements into which these may be subdivided.

The base tag set for verse extends the facilities provided by these two tag sets in the following ways:

- "numbered" <lg> elements are provided, by analogy with the "numbered" *div*n class elements discussed in section 7.1 ('Divisions of the Body') on p. 185 (see section 9.2 ('Structural Divisions of Verse Texts') on p. 214)
- a special purpose <caesura> element is provided, to allow for segmentation of the verse line (see section 9.3 ('Components of the Verse Line') on p. 218)
- a set of attributes is provided for the encoding of rhyme scheme and metrical information (see sections 9.4 ('Rhyme and Metrical Analysis') on p. 221 and 9.5 ('Rhyme') on p. 225)

To enable the base tag set for verse texts, a parameter entity *TEI.verse* must be declared within the document type subset, the value of which is "INCLUDE", as further described in section 3.3 ('Invocation of the TEI DTD') on p. 39. A document using this base tag set and no additional tag sets will thus begin as follows:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
     <!ENTITY % TEI.verse 'INCLUDE' >
]>
```

This declaration makes available the elements and attributes described in this chapter, in addition to those described in chapters 6 ('Elements Available in All TEI Documents') on p. 119 and 7 ('Default Text Structure') on p. 183.

9.1 Structure of the Base Tag Set for Verse

The base tag set for verse contains the following entity declarations:

```
<!-- 9.1:  Base Tag Set for Verse: entities                 -->
<!-- Text Encoding Initiative: Guidelines for Electronic    -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.  -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy  -->
<!-- in any form is granted, provided this notice is        -->
<!-- included in all copies.                                -->
```

```
<!-- These materials may not be altered; modifications to     -->
<!-- these DTDs should be performed as specified in the       -->
<!-- Guidelines in chapter "Modifying the TEI DTD."           -->

<!-- These materials subject to revision. Current versions    -->
<!-- are available from the Text Encoding Initiative.         -->

<!-- First, declare the class of components specific to verse -->

<!ENTITY % x.comp.verse ''                                     >
<!ENTITY % m.comp.verse '%x.comp.verse lg1'                    >
<!ENTITY % mix.verse '| %m.comp.verse'                         >

<!-- Next define attributes common to metrical elements
     -->

<!ENTITY % a.metrical '
          met               CDATA              %INHERITED
          real              CDATA              #IMPLIED
          rhyme             CDATA              #IMPLIED'     >
<!ENTITY % a.enjamb '
          enjamb            CDATA              #IMPLIED'     >
<!-- Now, we define the class of elements which can appear    -->
<!-- only within metrical elements                            -->

<!ENTITY % x.phrase.verse ''                                   >
<!ENTITY % m.phrase.verse '%x.phrase.verse caesura'            >
```

The base tag set for verse contains the following element declarations:

```
<!-- 9.1:  Base Tag Set for Verse                             -->
<!-- First, declare the default text structure, which is the  -->
<!-- same for verse as it is for other kinds of text. (But    -->
<!-- declare it only if verse is the only base.)              -->

<![ %TEI.singleBase [
<!ENTITY % TEI.structure.dtd system 'teistr2.dtd'              >
%TEI.structure.dtd;
]]&nil;>


<!-- Finally, declare the elements specific to this base tag  -->
<!-- set                                                      -->

<!-- ... declarations from section 9.2                       -->
<!--     (Numbered line groups)                               -->
<!--     go here ...                                          -->
<!-- ... declarations from section 9.3                       -->
<!--     (Caesura)                                            -->
<!--     go here ...                                          -->
```

9.2 Structural Divisions of Verse Texts

Like other kinds of text, texts written in verse may be of widely differing lengths and structures. A complete poem, no matter how short, may be treated as a free-standing text, and encoded in the same way as a distinct prose text. A group of poems functioning as a single unit may be encoded either as a <group> or as a <text>, depending on the encoder's view of the text. For further discussion, including an example encoding for a verse anthology, see chapter 7 ('Default

Text Structure') on p. 183.

Many poems consist only of ungrouped lines. This short poem by Emily Dickinson is a simple case:

```
<text>
<front><head>1755</head></front>
<body>
<l>To make a prairie it takes a clover and one bee,</l>
<l>One clover, and a bee,</l>
<l>And revery.</l>
<l>The revery alone will do,</l>
<l>If bees are few.</l>
</text>
```

Often, however, lines are grouped, formally or informally, into stanzas, verse paragraphs, etc. The <lg> element defined in the core tag set (in section 6.11.1 ('Core Tags for Verse') on p. 177) may be used for all such groupings. It may thus serve for informal groupings of lines such as those of the following example from Allen Ginsberg:

```
<text><body><head>My Alba</head>

<lg type=free>
<l>Now that I've wasted</l>
<l>five years in Manhattan</l>
<l>life decaying</l>
<l>talent a blank</l>
</lg>
<lg>
<l>talking disconnected</l>
<l>patient and mental</l>
<l>sliderule and number</l>
<l>machine on a desk</l>
</lg>
<!-- ... -->
</text>
```

It may also be used to mark the verse paragraphs into which longer poems are often divided, as in the following example from Samuel Taylor Coleridge's *Frost at Midnight*:

```
<lg type=para>
<l>The Frost performs its secret ministry,
<l>Unhelped by any wind....
<!-- ... -->
<l>Whose puny flaps and freaks the idling Spirit
<l>By its own moods interprets, every where
<l>Echo or mirror seeking of itself,
<l part=i>And makes a toy of Thought.
</lg><lg>
<l part=f>But O! how oft,
<l>How oft, at school, with most believing mind
<l>Presageful, have I gazed upon the bars,
<l>To watch that fluttering <hi>stranger</hi>! ...
<!-- ... -->
</lg>
<lg>
<l>Dear Babe, that sleepest cradled by my side,
<!-- ... -->
```

Note, in the above example, the use of the part attribute on the <l> element, where a verse line is broken between two line groups, as discussed in section 6.11.1 ('Core Tags for Verse') on p. 177. (Note also that here and in some other examples in this chapter the end-tags for the <l> element, being redundant, have been omitted, as is allowed when the SGML feature OMITTAG is enabled.)

Most typically, however, the <lg> element is used to mark the highly regular line groups which characterize stanzaic and similar verse forms, as in the following example from Chaucer:

```
<lg>
<l>Sire Thopas was a doghty swayn;
<l>White was his face as payndemayn,
<l>His lippes rede as rose;
<l>His rode is lyk scarlet in grayn,
<l>And I yow telle in good certayn,
<l>He hadde a semely nose.
</lg><lg>
<l>His heer, his ber was lyk saffroun,
<l>That to his girdel raughte adoun;
<!-- ... -->
</lg>
```

Like other text-division elements, <lg> elements may be nested hierarchically. For example, one particularly common English stanzaic form consists of a quatrain or sestet followed by a couplet. The <lg> element may be used to encode both the stanza and its components, as in the following example from Byron:

```
<lg type=stanza>
   <lg type=sestet>
     <l>In the first year of Freedom's second dawn
     <l>Died George the Third; although no tyrant, one
     <l>Who shielded tyrants, till each sense withdrawn
     <l>Left him nor mental nor external sun:
     <l>A better farmer ne'er brushed dew from lawn,
     <l>A worse king never left a realm undone!
   </lg>
   <lg type=couplet>
     <l>He died &dash; but left his subjects still behind,
     <l>One half as mad &dash; and t'other no less blind.
   </lg>
</lg>
```

Note the use of the type attribute to name the type of unit encoded by the <lg> element; this attribute is common to all members of the *divn* class (see section 7.1.1 ('Un-numbered Divisions') on p. 185).[1] "Sestet" and "couplet" might conceivably also be used as the values of the met attribute in a metrical analysis, for which see below, section 9.4 ('Rhyme and Metrical Analysis') on p. 221. The type attribute is intended solely for conventional names of different classes of text block; the met attribute is intended for systematic metrical analysis.

The above example uses "un-numbered" line groups which can nest within each other to any depth. When the base tag set for verse is in use, "numbered" line groups may also be used as an alternative.

<lg1> contains a first-level (i.e. largest) group of verse lines functioning as a formal unit e.g. a stanza, refrain, verse paragraph, etc.

<lg2> contains a second-level (i.e. second largest) group of verse lines functioning as a formal unit e.g. a stanza, refrain, verse paragraph, etc.

The base tag set for verse defines up to five such numbered line group elements, <lg1> to <lg5> inclusive. These function in exactly the same way as the numbered *divn* class elements discussed in section 7.1 ('Divisions of the Body') on p. 185.

As an example of their use, consider the Shakespearean sonnet. This may be divided into two parts: a concluding couplet, and a body of twelve lines, itself subdivided into three quatrains:

```
<text><body>
<lg1 type=body>
<lg2 type=quatrain>
```

[1] For discussion of other attributes of this class, see 7.1.4 ('Partial and Composite Divisions') on p. 189.

```
<l>My Mistres eyes are nothing like the Sunne,
<l>Currall is farre more red, then her lips red
<l>If snow be white, why then her brests are dun:
<l>If haires be wiers, black wiers grown on her head:
</lg2>
<lg2>
<l>I have seene Roses damaskt, red and white,
<l>But no such Roses see I in her cheekes,
<l>And in some perfumes is there more delight,
<l>Then in the breath that from my Mistres reekes.
</lg2>
<lg2>
<l>I love to heare her speake, yet well I know,
<l>That Musicke hath a farre more pleasing sound:
<l>I graunt I never saw a goddesse goe,
<l>My Mistres when shee walkes treads on the ground.
</lg2>
</lg1>
<lg1 type=couplet>
<l>And yet by heaven I think my love as rare,
<l>As any she beli'd with false compare.
</lg1>
</text>
```

Particularly lengthy poetic texts are often subdivided into units larger than stanzas or paragraphs, which may themselves be subdivided. Spenser's *Faery Queene*, for example, consists of twelve "books" each of which contains a prologue followed twelve "cantos". Each prologue and each canto consists of nine-line "stanzas", each of which follows the same regular pattern. Other examples in the same tradition are easy to find.

Large structures of this kind are most conveniently represented by the *divn* class elements such as or <div1> described in section 7.1 ('Divisions of the Body') on p. 185. Thus the start of the *Faery Queene* might be encoded as follows:

```
<body>
<div1 type=book n='I'>
  <div2 type=canto n='1'>
     <lg type=stanza n='I.1.1'>
       <l>A noble knight was pricking on the plain
       <l>Ycladd in mightie armes and silver shielde,
  <!-- ... -->
</body>
```

The encoder must choose at which point in the hierarchy of structural units to introduce <lg> elements rather than a yet smaller element: it would (for example) also be possible to encode the above example as follows:

```
<body>
<div1 type=book n=I>
  <div2 type=canto n=I.1>
     <div3 type=stanza n=I.1.1>
       <l>A noble knight was pricking on the plain
       <l>Ycladd in mightie armes and silver shielde,
       <!-- ... -->
</body>
```

One reason for preferring the former version is that not all of Spenser's stanzaic verse is organized into both cantos and books. In a corpus containing other works as well as the *Faery Queene*, it would be inconvenient for stanzas to appear in one part as <div3> elements, and in another as <div2> elements.[2]

The numbered line group elements have the following formal definition:

[2]Another way of avoiding this problem would be to use un-numbered elements; see further 7.1 ('Divisions of the Body') on p. 185.

```
<!-- 9.2:  Numbered line groups                                  -->
<!ELEMENT lg1           - o  (head?, (l | lg2)+)                 >
<!ATTLIST lg1                %a.global;
                             %a.divn;
                             %a.metrical;                        >
<!ELEMENT lg2           - o  (head?, (l | lg3)+)                 >
<!ATTLIST lg2                %a.global;
                             %a.divn;
                             %a.metrical;                        >
<!ELEMENT lg3           - o  (head?, (l | lg4)+)                 >
<!ATTLIST lg3                %a.global;
                             %a.divn;
                             %a.metrical;                        >
<!ELEMENT lg4           - o  (head?, (l | lg5)+)                 >
<!ATTLIST lg4                %a.global;
                             %a.divn;
                             %a.metrical;                        >
<!ELEMENT lg5           - o  (head?, l+)                         >
<!ATTLIST lg5                %a.global;
                             %a.divn;
                             %a.metrical;                        >
<!-- This fragment is used in sec. 9.1                          -->
```

9.3 Components of the Verse Line

It is often convenient for various kinds of analysis to encode subdivisions of verse lines. The general purpose `<seg>` element defined in the tag set for segmentation and alignment (section 14.3 ('Segments and Anchors') on p. 355) is provided for this purpose:

`<seg>` contains any arbitrary phrase-level unit of text (including other `<seg>` elements). Attributes include:

part specifies whether or not the segment is complete. Legal values are:

- ***Y*** the segment is incomplete
- ***N*** either the segment is complete, or no claim is made as to its completeness
- ***I*** the initial part of an incomplete segment
- ***M*** a medial part of an incomplete segment
- ***F*** the final part of an incomplete segment

To use this element together with the base tag set for verse, the tag set for segmentation and alignment must also be enabled, by an additional declaration in the document type subset, as further described in section 3.3 ('Invocation of the TEI DTD') on p. 39. A document using the base tag set for verse and this additional tag sets will thus begin as follows:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
<!ENTITY % TEI.verse 'INCLUDE' >
<!ENTITY % TEI.linking 'INCLUDE' >
]>
```

In Old and Middle English alliterative verse, individual verse lines are typically split into half lines. The `<seg>` element may be used to mark these explicitly, as in the following example from Langland's *Piers Plowman*:

```
<l><seg>In a somer seson,</seg>
   <seg>whan softe was the sonne,</seg></l>
<l><seg>I shoop me into shroudes</seg>
   <seg>as I a sheep were,</seg></l>
<l><seg>In habite as an heremite </seg>
   <seg>unholy of werkes,</seg></l>
<l><seg>Went wide in this world </seg>
```

```
      <seg>wondres to here.</seg></l>
   <!-- ... -->
```

The <seg> element can be nested hierarchically, in the same way as the <lg> element, down to whatever level of detailed structure is required. In the following example, the line has been divided into *feet*, each of which has been further subdivided into into syllables. (The eccentric formatting ensures that no spaces are introduced within words unless they are within SGML tags and thus not significant.)

```
      <l><seg type=foot>
            <seg type=syll>Ar/><seg
                 type=syll>ma</>
            <seg type=syll>vi/></seg><seg
         type=foot><seg
                 type=syll>rum/><seg
                 type=syll>que</>
            <seg type=syll>ca/></seg><seg
         type=foot><seg
                 type=syll>no</>
            <seg type=syll>Tro/></seg><seg
         type=foot><seg
                 type=syll>iae</>
            <seg type=syll>qui</>
         </seg>
         <seg type=foot>
            <seg type=syll>pri/><seg
                 type=syll>mus</>
            <seg type=syll>ab</>
         </seg>
         <seg type=foot>
            <seg type=syll>or/><seg
                 type=syll>is</>
         </seg>
   </l>
```

The <seg> element may be used to identify any subcomponent of a line which has content; its type attribute may characterize such units in any way appropriate to the needs of the encoder. For the specific case of labeling each foot with its formal type ("dactyl", "spondee", etc.), and each syllable with its metrical or prosodic status (syllables bearing primary or secondary stress, long syllables, short syllables), however, the specialized attributes met and real are defined, which provide a more systematic framework than the type attribute; see section 9.4 ('Rhyme and Metrical Analysis') on p. 221 below.

In classical verse, a hexameter like that above may also be formally divided into two "hemistiches". This example provides an interesting case, in that the boundary of the first hemistich falls in the middle of one of the feet (between the syllables "no" and "Tro"). If both kinds of segmentation are required, the part attribute might be used to mark the overlapping structure as follows. (This example uses the slightly artificial convention that word space is included within the syllable segments, and all other white space is to be ignored; though convenient for visual clarity in presentations like this one, this convention is not recommended for normal use.)

```
      <l><seg type=hemistich>
            <seg type=foot>
              <seg type=syll> Ar</>
              <seg type=syll>ma </>
              <seg type=syll>vi</>
            </seg>
            <seg type=foot>
              <seg type=syll>rum</>
              <seg type=syll>que </>
              <seg type=syll>ca</>
              </seg>
```

```
      <seg type=foot part=I>
        <seg type=syll>no </>
      </seg>
    </seg>
    <seg type=hemistich>
      <seg type=foot part=F>
        <seg type=syll>Tro</>
      </seg>
      <seg type=foot>
         <seg type=syll>iae </>
         <seg type=syll>qui </>
      </seg>
    <!-- ... -->
    </seg>
```

Numbered <seg> elements may also be used; these have the disadvantage—or advantage—of requiring a consistent, layer-by-layer analysis, and the advantage of providing a useful default for the type attribute, which must be specified only once for each level, as well as allowing the omission of many end-tags, which makes the presentation more compact.

```
<l><seg1 type=hemistich>
      <seg2 type=foot>
            <seg3 type=syll> Ar<seg3>ma <seg3>vi</>
      <seg2><seg3>rum<seg3>que <seg3>ca</>
      <seg2 part=I><seg3>no </>
   <seg1>
      <seg2 part=F><seg3>Tro</>
      <seg2><seg3>iae <seg3>qui </>
      <!-- ... -->
 </l>
<!-- ... -->
```

Instead of using the part attribute on the <seg2> element, it might be simpler just to mark the point at which the caesura occurs. An additional element is provided for analyses of this kind, in which what is to be marked are points "between the words", which have some significance within a verse line:

<caesura> marks the point at which a metrical line may be divided.

In classical prosody, the *caesura*, which occurs within a foot, is distinguished from a *diaeresis*, which occurs on a foot boundary (not to be confused with the division of a diphthong into two syllables, or the diacritic symbol used to indicate such division, each of which is also termed *diaeresis*). This distinction is rarely made nowadays, the term 'caesura' being used for any division irrespective of foot boundaries. No special-purpose <diaeresis> element is therefore provided.

As an example of the <caesura> element, we refer again to the example from Langland. An encoder might choose simply to record the location of the caesura within each line, rather than encoding each half-line as a segment in its own right, as follows:

```
<l>In a somer seson, <caesura> whan softe was the sonne, </l>
<l>I shoop me into shroudes <caesura> as I a sheep were, </l>
<l>In habite as an heremite <caesura> unholy of werkes, </l>
<l>Went wide in this world <caesura> wondres to here. </l>
```

Logically, the opposite of caesura might be considered to be enjambement. The base tag set for verse defines an enjamb attribute for the <l> element, which alllows the presence or absence of enjambement to be signaled. The following lines demonstrate the use of the enjamb attribute to mark places where there is a discrepancy between the boundaries of the <l> elements and the syntactic structure of the verse (a discrepancy of some significance in some schools of verse):

```
<l enjamb=y>Un astrologue, un jour, se laissa choir</l>
<l>Au fond d'un puits.
```

The elements discussed in this section are formally defined as follows:

```
<!-- 9.3:  Caesura                                          -->
<!ELEMENT caesura       - O  EMPTY                          >
```

```
<!ATTLIST caesura            %a.global;                          >
<!-- This fragment is used in sec. 9.1                         -->
```

9.4 Rhyme and Metrical Analysis

When the base tag set for verse is in use, the following additional attributes are available to record information about rhyme and metrical form:

met contains a user-specified encoding for the conventional metrical structure of the element.

real contains a user-specified encoding for the actual realization of the conventional metrical structure applicable to the element.

rhyme specifies the rhyme scheme applicable to a group of verse lines.

These attributes may be attached to the <lg> element, to any of its numbered equivalents <lg1>, <lg2>, etc., to the higher-level text-division elements , <div1>, etc., or to the <body> element itself if the whole of a text is in verse. In general, the attributes should be specified at the highest level possible; they may not however be specifiable at the highest level if some of the subdivisions of a text are in prose and others in verse. All these attributes may also be attached to the <l> and <seg> elements, but the default notation for the rhyme attribute has no defined meaning when specified on <l> or <seg>. The value for these attributes may take any form desired by the encoder; the nature of the notation used will determine how well the attribute values can be processed by automatic means.

The primary function of the metrical attributes is to encode the conventional metrical or rhyming structure within which the poet is working, rather than the actual prosodic realization of each line; the latter can be recorded using the real attribute, as further discussed below. There is no provision at this time, however, for encoding the particular realization of a rhyme pattern.

9.4.1 Sample Metrical Analyses

As a simple example of the use of these attributes, consider the following lines from Pope's *Essay on Criticism*:

```
<text>
  <front> ... </front>
  <body met='-+|-+|-+|-+|-+/' rhyme='aa'>
    <lg1 n=1 type="verse paragraph">
      <l>'Tis hard to say, if greater Want of Skill</l>
      <l>Appear in <hi>Writing</> or in <hi>Judging</hi> ill;</l>
      <l>But, of the two, less dang'rous is th'Offence,</l>
      <l>To tire our <hi>Patience</hi>, than mis-lead
         our <hi>Sense</hi>:</l>
    <!-- ... -->
  </body>
</text>
```

The body of this text is written entirely in *heroic couplets*; each line is iambic pentameter (which, using a common notation, can be described with the formula `-+|-+|-+|-+|-+/`, each `-` denoting a metrically unstressed syllable, each `+` a metrically stressed one, each `|` a foot boundary, and the `/` a line-end), and the couplets rhyme (which can be represented with the conventional formula `aa`).

Because both rhyme pattern and metrical form are consistent throughout the poem, they may be most conveniently specified on the <body> element; the values given for the attributes will be inherited by any metrical unit contained within this <body> element, and must be interpreted in the appropriate way.

Since the notation used in the met, real, and rhyme attributes is user-defined, no binding description can be given of its details or of how its interpretation must proceed. (A default notation is provided for the rhyme attribute, which however the encoder can replace with

another; see section 9.5 ('Rhyme') on p. 225.) It is expected, however, that software should be able to support these attributes in useful ways; the more intelligent the software is, and the more knowledge of metrics is built into it, the better it will be able to support these attributes. In the extract given above, for example, the met and rhyme attribute values specified on the <body> element are inherited directly by the <lg> elements nested within the body. Since the met value specifies the metrical form of a single verse line, the structure of the <lg> as a whole is understood to involve as many repetitions of the pattern as there are lines in the verse paragraph. The same attribute value, when inherited in turn by the <l> element, must be understood *not* to repeat. With sufficiently sophisticated software, segments within the line might even be understood as inheriting precisely that portion of the formula which applies to the segment in question; this will, however, be easier to accomplish for some languages than for others.

The rhyme attribute in this example uses the default notation to specify a rhyme scheme applicable only to pairs of lines. As elsewhere, the default notation for the rhyme attribute has no meaning for metrical units at the line level or below. In verse forms where line-internal rhyme is structurally significant, e.g. in some skaldic poetry, the default notation is incapable of expressing the required information, since the rhyme pattern may need to be specified for units smaller than the line. In such cases, a user-specified rhyme notation must be substituted for the default notation, or else the rhyme pattern must be described using some alternative method (e.g. by using the <link> mechanism described below).

The precise semantics of the met attribute and the inferences which software is expected or able to draw from it, are implementation-dependent; so are the semantics and processing of the rhyme attribute, when user-specified notations are used.

A formal definition of the significance of each component of the pattern given as the value of the met attribute may be provided in the <metDecl> element within the <encodingDecl> element in the TEI header (see section 5.3.8 ('The Metrical Declaration Element') on p. 106). The encoder is free to invent any notation appropriate to his or her analytic needs, provided that it is adequately documented in this element. The notation may define metrical components using invented or traditional names (such as "iamb" or "hexameter") or in terms of basic units such as codes for stressed or unstressed syllables, or a combination of the two.

The real (for "realization") attribute may optionally be specified to indicate any deviation from the pattern defined by the met attribute which the encoder wishes to record. By default, the real attribute has the same value as the met attribute on the same element; it is only necessary to provide an explicit value when the realization differs in some way from the abstract metrical pattern. The tension between conventional metrical pattern and its realization may thus be recorded explicitly. For example, many readers of the above passage would stress the word "But" at the beginning of the third line rather than the word "of" following it, as the metrical pattern would normally require. This variation might be encoded as follows:

```
<l real='+-|-+|-+|-+|-+'>But of the two...
```

Where the real attribute is used to over-ride the default or conventional metrical pattern, it applies only to the element on which it is specified. The default pattern for any subsequent lines is unaffected.

As it happens, this particular kind of variation is very common in the English iambic pentameter;[3] an encoder might therefore choose to regard this not as an instance of a variant realization, but as an instance of a variant metrical form:

```
<l met='+-|-+|-+|-+|-+'>But of the two...
```

Alternatively, a different metrical notation might be defined, in which this kind of variation was permitted throughout the text.

In choosing whether to over-ride a metrical specification in this way or by using the real attribute, the encoder is required to determine whether the change is a systematic or conventional one (as in this example) or an occasional variation, perhaps for local effect. In the following example, from Goethe's *Auf dem See*, the variation is a matter of local realization:

```
<lg1 type="chevy-chase-stanza"
```

[3] It even has a name: *anapaestic substitution*.

```
      met="-+-+-+-+/-+-+-+"
      rhyme='ababcdcd'>
  <l n=1>                  Und frische Nahrung, neues Blut
  <l n=2 real="+--+-+">    Saug' ich aus freier Welt;
  <l n=3 real="+--+-+-+">  Wie ist Natur so hold und gut,
  <l n=4 real="---+-+">    Die mich am Busen haelt!
  <l n=5>                  Die Welle wieget unsern Kahn
  <l n=6>                  Im Rudertakt hinauf,
  <l n=7>                  Und Berge, wolkig himmelan,
  <l n=8>                  Begegnen unserm Lauf.
```

On the other hand, the famous inserted alexandrine in Pope's "Essay on Criticism", might be encoded as follows:

```
<l n=356> A needless alexandrine ends the song,
<l n=357 met='-+|-+|-+|-+|-+|-+' real='++|-+|-+|+-|++|-+'>
That, like a wounded snake, drags its slow length along.
</l>
```

Here the met attributes indicates that a different metrical convention (the alexandrine) is in force, while the real attribute indicates that there is a variation from that convention. As with many other aspects of metrical analysis, however, this is of necessity an entirely interpretive judgment.

9.4.2 Segment-Level versus Line-level Tagging

The examples given so far have encoded information about the realization of metrical conventions at the level of the whole verse-line. This has obvious advantages of simplicity, but the disadvantage that any deviation from metrical convention is not marked at its precise point of occurrence in the text. Greater precision may be achieved, but only at the cost of marking deviant metrical units explicitly. This may be done with the <seg> element, giving the variant realization as the value of the real attribute on that element. Using this method, the example given immediately above might be encoded as follows:

```
<l n=356> A need<seg1 type=foot n=2 real='--'>less
   a</seg1>lexandrine ends the song,
<l n=357 met='-+|-+|-+|-+|-+|-+'>
   <seg1 n=1 real='++'> That, like </seg1> a wounded snake,
   <seg1 n=4 real='++'> drags its </seg1>
   <seg1 n=5 real='++'> slow length </seg1>
   along.
</l>
```

The marking of the foot boundaries with the symbol | in the met attribute value of the <l> element allows the human reader, or a sufficiently intelligent software program, to isolate the correct portion of that attribute value as the default value for the same attribute on the <seg1> elements for feet, namely -+. It is of course up to the encoder to decide whether or not to include the n attribute of <seg> here, and whether or not also to tag the feet in the line in which there is no deviation from the metrical convention. The ability of software to infer which foot is being marked, if not all are tagged, will depend heavily on the language of the text and the knowledge of prosody built into the software; the fuller and more explicit the markup, the easier it will be for software to handle it. It may prove useful, however, to mark metrical deviations in the manner shown, even if the available software is not sufficiently intelligent to scan lines without aid from the markup. Human readers who are interested in prosody may well be able to exploit the markup in useful ways even with less sophisticated software.

There are circumstances where it may also be useful to use the met attribute of <seg>. If we wish to identify the exact location of the different types of foot in the first line of Virgil's *Aeneid*, the text could be encoded as follows (for simplicity's sake the caesura has been omitted):

```
<l><seg1 type=foot met='+--'>Arma vi</>
   <seg1 met='+--'>rumque ca</>
   <seg1 met='++'>no Tro</>
```

```
    <seg1 met='++'>iae qui </>
    <seg1 met='+--'>primus ab</>
    <seg1 met='++'> oris</>
</l>
```

An appropriate value of the met attribute might also be encoded on the enclosing or <body> element, to indicate that each foot may be made up of a dactyl or a spondee, so that the values given here for met at the level of the foot may be considered a series of local variations on this fundamental pattern; in cases like this, of course, the local variations may also be considered aspects of realization rather than of convention, in which case the real attribute may be used instead of met, if desired.

9.4.3 Metrical Analysis of Stanzaic Verse

The method described above may be used to encode quite complex verse forms, for instance various kinds of fixed-form stanzas. Let us take one of Dante's canzoni, in which each stanza except the last has the same combination of eleven-syllable and seven-syllable lines, and the same rhyme scheme:

```
<div0 type=canzone
      met='E/E/S/E/S/E/E/S/E/S/E/S/S/E/S/E/E/S/S/E/E'
      rhyme="abbcdaccbdceeffghhhgg">
<lg1 n=1 type=stanza>
<l n=1>Doglia mi reca nello core ardire
  <!-- ... -->
```

Here the met attribute specifies a metrical pattern for each of the twenty-one lines making up a stanza of the *canzone*. Each stanza inherits this definition from the parent <div0> element. The rhyme attribute specifies a rhyme scheme for each stanza, in the same way.

In the metrical notation used here, the letter E represents a line containing nine syllables which may or may not be metrically prominent, a tenth which is prominent and an optional non-prominent eleventh syllable. The letter "S" is used to represent a line containing five syllables which may or may not be metrically prominent, a sixth which is prominent and an optional non-prominent seventh syllable. A suitable definition for this notation might be given by a <metDecl> element like the following:

```
<metDecl type='met' pattern='((E|S)/)+)'>
  <symbol value='E' terminal=n>xxxxxxxxx+o
  <symbol value='S' terminal=n>xxxxx+o
  <symbol value='x'>metrically prominent or non-prominent
  <symbol value='+'>metrically prominent
  <symbol value='o'>optional non prominent
  <symbol value='/'>line division
</metDecl>
```

As noted above, the metrical pattern specified on the <div0> applies to each <lg> (stanza) element contained within the <div0>. In fact however, after seven stanzas of this type, there is a final stanza, known as a *commiato* or envoi, which follows a different metrical and rhyming scheme. The solution to this problem is simply to specify a new met attribute on the eighth stanza itself, which will override the default value inherited from parent <div0>, as follows:

```
<div0 met='.....'>
<!-- ... -->
<lg1>
  <!-- This line group inherits its met value -->
  <!-- from the containing <div0>             -->
  <l> ...
</lg1>
<lg1 type='commiato'
    met='E/S/S/E/S/E/E/S/S/E/E'
    rhyme="abbccdeeedd">
  <l n=1> Canzone, presso di qui
   <!-- ... -->
```

```
</lg1>
</div0>
```

Note that, in the same way as for the real attribute, over-riding of this kind does not affect subsequent elements at the same hierarchic level. Any <lg1> element following the *commiato* above would be assumed to use the same metrical and rhyming scheme as the one preceding the *commiato*. Moreover, although it is quite regular (in the sense that the last stanza of each *canzone* is a *commiato*), the over-riding must be specified for each case.

9.5 Rhyme

The rhyme attribute is used to specify the rhyme pattern of a verse form. Like the met attribute, it can be used with a user-specified notation documented by the <metDecl> element in the TEI header. Unlike met, however, the rhyme attribute has a default notation; if this default notation is used, no <metDecl> element need be given.

The default notation for rhyme offers the ability to record patterns of rhyming lines, using the traditional notation in which distinct letters stand for rhyming lines. For a work in rhyming couplets, like the Pope example above, the rhyme attribute simply specifies aa, indicating that pairs of adjacent lines rhyme with each other. For a slightly more complex scheme, applicable to groups of four lines, in which lines 1 and 3 rhyme, as do lines 2 and 4, this attribute would have the value abab. The traditional Spenserian stanza has the pattern ababbcbcc, indicating that within each nine line stanza, lines 1 and 3 rhyme with each other, as do lines 2, 4, 5 and 7, and lines 6, 8 and 9.

Non-rhyming lines within such a group may be represented using a hyphen or an x, as in the following example:

```
<lg rhyme='AB-BBA'>
<l>The sunlight on the garden
<l>Hardens and grows cold,
<l>We cannot cage the minute
<l>Within its nets of gold
<l>When all is told
<l>We cannot beg for pardon.
</lg>
```

Note however that the default notation includes no specific way of recording "internal" rhyme, such as that between the end of the first line and the start of the second in this particular poem. For this, a special user-defined notation would need to be declared using the <metDecl> element in the header. Alternatively, rhyme, like alliteration or assonance, may be considered as a special form of "correspondence", and hence encoded using the mechanisms defined for that purpose in section 14.4 ('Correspondence and Alignment') on p. 358.

To use the correspondence mechanisms to represent the complex rhyming pattern of the above example, we first need to delimit and identify each rhyming sequence within the text: the <seg> element may be used for this purpose, as follows:

```
<lg rhyme='AB-BBA'>
<l>The sunlight on the <seg id=A1>garden</seg>
<l><seg id=A2>Harden</seg>s and grows <seg id=B1>cold,</seg>
<l>We cannot cage the <seg id=C1>minute</seg>
<l>Wi<seg id=C2>thin it</seg>s nets of <seg id=B2>gold</seg>
<l>When all is <seg id=B3>told</seg>
<l>We cannot beg for <seg id=A3>pardon</seg>.
</lg>
```

Now that each rhyming word, or part-word, has been tagged and allocated an arbitrary identifier, the general purpose <link> element may be used to indicate which of the <seg> elements share the same rhyme, as follows:

```
<linkGrp type=rhyme>
<link targets='A1 A2 A3'>
<link targets='B1 B2 B3'>
<link targets='C1 C2'>
</linkGrp>
```

For further discussion of the <link> and <linkGrp> element, see section 14.4 ('Correspondence and Alignment') on p. 358.

9.6 Encoding Procedures For Other Verse Features

A number of procedures that may be of particular concern to encoders of verse texts are dealt with elsewhere in these guidelines. Some aspects of layout and physical appearance, especially important in the case of free verse, are dealt with in chapter 18 ('Transcription of Primary Sources') on p. 443. Some initial recommendations for the encoding of phonetic or prosodic transcripts, which may be helpful in the analysis of sound structures in poetry, are to be found in chapter 11 ('Transcriptions of Speech') on p. 249; it may also be found convenient to use standard entity names (those proposed for the International Phonetic Alphabet suggest themselves) to mark positions of suprasegmentals such as primary and secondary stress, or other aspects of accentual structure.

As already indicated, chapter 14 ('Linking, Segmentation, and Alignment') on p. 331 contains much which will be found useful for the aligning of multiple levels of commentary and structure within verse analysis. Encoders of verse (as of other types of literary text) will frequently wish to attach identifying labels to portions of text that are not part of a system of hierarchical divisions, may overlap with one another, and/or may be discontinuous; for instance passages associated with particular characters, themes, images, allusions, topoi, styles, or modes of narration. Much of the computerized analysis of verse seems likely to require dividing texts up into blocks in this way. The element discussed in 15.3 ('Spans and Interpretations') on p. 387 provides the means for doing this. Finally, the procedures for the tagging of feature structures, described in chapter 16 ('Feature Structures') on p. 397, provide a powerful means of encoding a wide variety of aspects of verse literature, including not only the metrical structures discussed above, but also such stylistic and rhetorical features as metaphor.

For other features it must for the time being be left to encoders to devise their own terminology. Elements such as <metaphor tenor= ... vehicle= ...> ... </metaphor> might well suggest themselves; but given the problems of definition involved, and the great richness of modern metaphor theory, it is clear that any such format, if pre-defined by these Guidelines, would have seemed objectionable to some and excessively restrictive to many. Leaving the choice of tagging terminology to individual encoders carries with it one vital corollary, however: the encoder must be utterly explicit, in the TEI header, about the methods of tagging used and the criteria and definitions on which they rest. Where no formal elements are currently proposed, such information may readily be given as simple prose description within the <encodingDesc> element defined in section 5.3 ('The Encoding Description') on p. 93.

Chapter 10

Base Tag Set for Drama

This base tag set is intended for use when encoding printed dramatic texts, screen plays or radio scripts, and written transcriptions of any form of performance.

Section 10.1 ('Front and Back Matter ') on p. 228 discusses elements, such as cast lists, which can appear only in the front or back matter of printed dramatic texts. Section 10.2 ('The Body of a Performance Text') on p. 235 discusses the structural components of performance texts: these include major structural divisions such as acts or scenes (section 10.2.1 ('Major Structural Divisions') on p. 236); individual speeches (section 10.2.2 ('Speeches and Speakers') on p. 237); stage directions (section 10.2.3 ('Stage Directions') on p. 238); and the elements making up individual speeches (section 10.2.4 ('Speech Contents') on p. 241). Section 10.2.5 ('Embedded Structures') on p. 242 discusses ways of encoding units which cross the simple hierarchic structure so far defined, such as embedded songs or other masques. Finally, section 10.3 ('Other Types of Performance Text') on p. 246 discusses a small number of additional elements characteristic of screen plays and radio or television scripts, as well as some elements for representing technical stage directions such as lighting or blocking.

To enable the base tag set for performance texts, the parameter entity *TEI.drama* must be declared within the document type subset with the value `INCLUDE`, as further described in section 3.3 ('Invocation of the TEI DTD') on p. 39. A document using the base tag set for drama and no additional tag sets will thus begin as follows:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
     <!ENTITY % TEI.drama 'INCLUDE' >
]>
```

This declaration makes available all of the elements described in this chapter, in addition to the core elements described in chapter 6 ('Elements Available in All TEI Documents') on p. 119. The default structure for dramatic texts is similar to that defined by chapter 7 ('Default Text Structure') on p. 183, as further discussed in section 10.2.1 ('Major Structural Divisions') on p. 236.

Three additional element classes are defined by this base tag set. The *dramafront* class contains elements which can appear only in the front or back matter of performance texts. The *stageDirection* class contains a set of elements for specialized stage directions, which can occur between or within speeches. The *comp.drama* class contains all elements which may appear as components of performance texts, in addition to those defined in the core. The classes are defined using parameter entities as shown below:

```
<!-- 10:  Class declarations for Performance Texts           -->
<!ENTITY % x.dramafront ''                                     >
<!ENTITY % m.dramafront '%x.dramafront epilogue | performance |
          prologue | set'                                      >
<!ENTITY % x.stageDirection ''                                 >
<!ENTITY % m.stageDirection '%x.stageDirection camera | caption
          | move | sound | tech | view'                        >
<!ENTITY % x.comp.drama ''                                     >
<!ENTITY % m.comp.drama '%x.comp.drama %m.stageDirection; |
```

```
            castList'                                              >
<!ENTITY % mix.drama '| %m.comp.drama'                             >
```

The remainder of the DTD fragment defining the base tag set for drama has the following overall shape:

```
<!-- 10:  Base tag set for Performance texts                    -->
<!-- ... declarations from section 10.1                         -->
<!--  (Specialized front and back matter for performance        -->
<!-- texts)                                                     -->
<!--     go here ...                                            -->
<!-- ... declarations from section 10.2.3                       -->
<!--     (Stage directions)                                     -->
<!--     go here ...                                            -->
<!-- ... declarations from section 10.3.1                       -->
<!--     (Screen plays and other technical matters)             -->
<!--     go here ...                                            -->

<!-- The base tag set for drama uses the standard default      -->
<!-- text-structure elements, which are embedded here:         -->


<![ %TEI.singleBase [
<!ENTITY % TEI.structure.dtd system 'teistr2.dtd'                  >
%TEI.structure.dtd;
]]&nil;>
```

10.1 Front and Back Matter

In dramatic texts, as in all TEI conformant documents, the header element is followed by a <text> element, which contains optional front and back matter, and either a <body> or else a <group> of nested <text> elements. For more information on these, see chapter 7 ('Default Text Structure') on p. 183.

The <front> and <back> elements are most likely to be of use when encoding preliminary materials in published performance texts. These often contain specific textual elements not generally found in other forms of text. These include:

<performance> contains a section of front or back matter describing how a dramatic piece is to be performed in general or how it was performed on some specific occasion.

<prologue> contains the prologue to a drama, typically spoken by an actor out of character, possibly in association with a particular performance or venue.

<epilogue> contains the epilogue to a drama, typically spoken by an actor out of character, possibly in association with a particular performance or venue.

<set> contains a description of the setting, time, locale, appearance, etc., of the action of a play, typically found in the front matter of a printed performance text (not a stage direction).

<castList> contains a single cast list or dramatis personae.

Elements for encoding each of these specific kinds of front matter are discussed in the remainder of this section, in the order given above. In addition, the front matter of dramatic texts may include the same elements as that of any other kind of text, notably title pages and various kinds of text division, as discussed in section 7.4 ('Front Matter') on p. 201. The encoder may choose to ignore the specialized elements discussed in this section and instead use constructions of the type or <div1 type=set>.

Most other material in the front matter of a performance text will be marked with the default text structure elements described in chapter 7 ('Default Text Structure') on p. 183. For example, the title page, dedication, other commendatory material, preface, etc., in a printed text should be encoded using or <div1> elements, containing headings, paragraphs, and other core tags.

The specialized elements for front and back matter of performance texts are defined as follows:

```
<!-- 10.1: Specialized front and back matter for performance  -->
<!-- texts                                                    -->
<!-- ... declarations from section 10.1.1                     -->
<!--     (The Set element)                                    -->
<!--     go here ...                                          -->
<!-- ... declarations from section 10.1.2                     -->
<!--     (Prologues and Epilogues)                            -->
<!--     go here ...                                          -->
<!-- ... declarations from section 10.1.3                     -->
<!--     (The Performance element)                            -->
<!--     go here ...                                          -->
<!-- ... declarations from section 10.1.4                     -->
<!--     (The cast list)                                      -->
<!--     go here ...                                          -->
<!-- This fragment is used in sec. 10                         -->
```

10.1.1 The Set Element

A special form of note describing the setting of a dramatic text (that is, the time and place of its action) is sometimes found in the front matter.

<set> contains a description of the setting, time, locale, appearance, etc., of the action of a play, typically found in the front matter of a printed performance text (not a stage direction).

Descriptions of the setting may also appear as initial stage directions in the body of the play, but such descriptions should be marked as stage directions, not <set>. The <set> element should be used only where the description forms part of the front matter, as in the following examples:

```
<front>
  <castlist>
     <!-- ... -->
  </castlist>
  <set>The action of the play is set in Chicago's
       Southside, sometime between World War II and the
       present.</set>
</front>
```

```
<front>
  <titlePage> ... </titlePage>
  <div type='copyright page'> ... </div>
  <div type='Contents'> ... </div>
  <div type='Introduction'> ... </div>
  <div type=''><head>Note on the Translation</head> ... </div>
  <titlePage type='half-title'>
  <docTitle><titlePart>Peer Gynt</></docTitle>
  </titlePage>
  <div type='Dramatis Personae'><head>Characters</> ... </div>
  <set>The action, which opens in the early years of the
       last century and closes about fifty years later,
       takes place partly in the Gudbrand Valley in Norway
       and on the mountains around it, partly on the
       Moroccan coast, partly in the Sahara Desert,
       the asylum in Cairo, at sea, etc.</set>
   <performance> ... </performance>
</front>
```

The <set> element is formally defined as follows:

```
<!-- 10.1.1:  The Set element                                 -->
<!ELEMENT set           - -  (head?, %specialPara)             >
```

```
<!ATTLIST set                %a.global;                          >
<!-- This fragment is used in sec. 10.1                        -->
```

10.1.2 Prologues and Epilogues

Many plays in the Western tradition include in their front matter a prologue, spoken by an actor, generally not in character. Similar speeches often also occur at the end of the play, as epilogues. The elements <prologue> and <epilogue> are provided for the encoding of such features within the front or back matter, where appropriate.

<prologue> contains the prologue to a drama, typically spoken by an actor out of character, possibly in association with a particular performance or venue.

<epilogue> contains the epilogue to a drama, typically spoken by an actor out of character, possibly in association with a particular performance or venue.

A prologue may be encoded just like a distinct poem, as in the following example:

```
<front>
<prologue>
<head>Prologue, spoken by <name>Mr. Hart</name></head>
<l>Poets like Cudgel'd Bullys, never do
<l>At first, or second blow, submit to you;
<l>But will provoke you still, and ne're have done,
<l>Till you are weary first, with laying on:
<!-- ... -->
<l>We patiently you see, give up to you,
<l>Our Poets, Virgins, nay our Matrons too.
</prologue>
<castlist>
<head>The Persons</head>
<!-- ... -->
</castlist>
<set><head>The SCENE</>
London</set>
</front>
```

A prologue or epilogue may also be encoded as a speech, using the <sp> element described in section 6.11.2 ('Core Tags for Drama') on p. 179. This is particularly appropriate where stage directions, etc., are involved, as in the following example:

```
<epilogue>
<head>Written by <name>Colley Cibber, Esq</name>
and spoken by <name>Mrs. Cibber</name></head>
<sp>
<lg>
<l>Since Fate has robb'd me of the hapless Youth,
<l>For whom my heart had hoarded up its truth;
<l>By all the Laws of Love and Honour, now,
<l>I'm free again to chuse, &mdash; and one of you
</lg>
<!-- ... -->
<lg><l>Suppose I search the sober Gallery; &mdash; No,
<l>There's none but Prentices &mdash; &amp; Cuckolds all a row:
<l>And these, I doubt, are those that make 'em so.</l></lg>
<stage>Pointing to the Boxes.</stage>
<lg>
<l>'Tis very well, enjoy the jest:
<!-- ... -->
</sp>
</epilogue>
```

In cases where the prologue or epilogue is clearly a significant part of the dramatic action, it may be preferable to include it in the body of a text, rather than in the front or back matter.

In such cases, the encoder (and theatrical tradition) will determine whether or not to regard it as a new scene or division, or simply the final speech in the play. In the First Folio version of Shakespeare's *Tempest*, for example, Prospero's final speech is clearly marked off as a distinct textual unit by the headings and layout of the page, and might therefore be encoded as back matter:

```
<sp who=PR>
<l part=y>I'le deliver all,
<l>And promise you calme Seas, auspicious gales,
<!-- ... -->
<l>Be free and fare thou well: please you, draw neere.</l>
<stage>Exeunt omnes.</stage>
</div1>
</body>
<back>
<epilogue>
<head>Epilogue, spoken by Prospero.</head>
<sp>
<l>Now my Charmes are all ore-throwne,
<l>And what strength I have's mine owne
<!-- ... -->
<l>As you from crimes would pardon'd be,
<l>Let your Indulgence set me free.<stage>Exit</stage>
</sp>
</epilogue>
<!>
<set>The Scene, an un-inhabited Island.</set>
<!>
<castList>
<head>Names of the Actors.</>
<castitem>Alonso, K. of Naples</>
<castItem>Sebastian, his Brother.</>
<castItem>Prospero, the right Duke of Millaine.</>
<!-- ... -->
</castList>
<trailer>FINIS</>
</back>
</text>
```

In many more modern editions, the editors have chosen to regard Prospero's speech as a part of the preceding scene:

```
<sp who=PR><speaker>Prospero</speaker>
<l part=y>I'll deliver all,
<l>And promise you calm seas, auspicious gales,
<!-- ... -->
<l>Be free and fare thou well. <stage>Exit Ariel</stage>
Please you, draw near.</l>
<stage>Exeunt all but Prospero</stage>
<note place=margin>Epilogue</note>
<l>Now my charms are all o'erthrown,
<l>And what strength I have's mine own
<!-- ... -->
<l>As you from crimes would pardoned be,
<l>Let your indulgence set me free.</l>
<stage>He awaits applause, then exit.</stage>
</sp>
```

Prologues and epilogues are formally defined as follows:

```
<!-- 10.1.2:  Prologues and Epilogues                          -->
<!ELEMENT prologue      - -  ((%m.divtop)*, (%component)+,
                              (%m.divbot)*)                     >
<!ATTLIST prologue           %a.global;                         >
```

```
<!ELEMENT epilogue      - -  ((%m.divtop)*, (%component)+,
                              (%m.divbot)*)                    >
<!ATTLIST epilogue           %a.global;                        >
<!-- This fragment is used in sec. 10.1                       -->
```

10.1.3 Records of Performances

Performance texts are not only printed in books to be read, they are also performed. It is common practice therefore to include within the front matter of a printed dramatic text some brief account of particular performances, using the following element:

<performance> contains a section of front or back matter describing how a dramatic piece is to be performed in general or how it was performed on some specific occasion.

The <performance> element may be used to group any and all information relating to the actual performance of a play or screenplay, whether it specifies how the play should be performed in general or how it was performed in practice on some occasion.

Performance information may include complex structures such as cast lists, or paragraphs describing the date and location of a performance, details about the setting portrayed in the performance and so forth. (See the discussion of these specialized structures in section 10.1 ('Front and Back Matter ') on p. 228 above.) If information for more than one performance is being recorded, then more than one <performance> element should be used.

Names of persons, places, and dates of particular significance within the performance record may be explicitly marked using the general purpose <name>, <rs type=place> and <date> elements described in section 6.4.4 ('Dates and times') on p. 137. No particular elements for such features as theatres, directors, etc., are proposed at this time.

For example:

```
<performance>
<head>Death of a Salesman</head>
<p>A New Play by Arthur Miller</>
<p>Staged by Elia Kazan</>
<castList>
<head>Cast</head>
<note place='inline' rend='small type flush left'>
(in order of appearance)
</note>
<castItem><role>Willy Loman</> <actor>Lee J. Cobb</></>
<castItem><role>Linda</>       <actor>Mildred Dunnock</></>
<castItem><role>Biff</>        <actor>Arthur Kennedy</></>
<castItem><role>Happy</>       <actor>Cameron Mitchell</></>
<!-- ... -->
</castList>
<p>The setting and lighting were designed by <name>Jo Mielziner</>.
<p>The incidental music was composed by <name>Alex North</>.
<p>The costumes were designed by <name>Julia Sze</>.
<p>Presented by <name rend=unmarked>Kermit Bloomgarden</> and
<name rend=unmarked>Walter Fried</> at the <rs type=place>Morosco
Theatre in New York</> on <date>February 10, 1949</>.
</performance>
```

Or:

```
<performance>
<p>La Machine Infernale a &eacute;t&eacute;
repr&eacute;sent&eacute;e pour la premi&egrave;re fois au
<rs type='place (theatre)'>th&eacute;&acirc;tre Louis-Jouvet</rs>
<rs type='place (theatre)'>(Com&eacute;die des
Champs-&Eacute;lys&eacute;es)</rs>
<date>le 10 avril 1934</date>,
avec les d&eacute;cors et les costumes de
<name>Christian B&eacute;rard.</name>
```

```
<!-- ... -->
</performance>
```

The <performance> element is formally defined as follows:

```
<!-- 10.1.3:  The Performance element                          -->
<!ELEMENT performance   - -  ((%m.divtop)*, (%component)+,
                             (%m.divbot)*)                      >
<!ATTLIST performance        %a.global;                         >
<!-- This fragment is used in sec. 10.1                        -->
```

10.1.4 Cast Lists

A *cast list* is a specialized form of list, conventionally found at the start or end of a play, usually listing all the speaking and non-speaking roles in the play, often with additional description ("Cataplasma, a maker of Periwigges and Attires") or the name of an actor or actress ("Old Lady Squeamish. Mrs Rutter"). Cast lists may be encoded with the general purpose **<list>** element described in section 6.7 ('Lists') on p. 149, but for more detailed work the following specialized elements are provided:

<castList> contains a single cast list or dramatis personae.
<castGroup> groups one or more individual **<castItem>** elements within a cast list.
<castItem> contains a single entry within a cast list, describing
either a single role or a list of non-speaking roles. Attributes include:

type characterizes the cast item. Sample values include:

role the item describes a single role.

list the item describes a list of non-speaking roles.

<role> the name of a dramatic role, as given in a cast list.
<roleDesc> describes a character's role in a drama.
<actor> Name of an actor appearing within a cast list.

Cast lists often have an internal structure of their own; it is quite usual to find, for example, nobility and commoners, or male and female roles, presented in different groups or sublists. Roles are also often grouped together by their function, for example:

Sons of Cato:

Portius
Marcus

A cast list relating to a specific performance may be accompanied by notes about the time or place of that performance, indicating (for example) the name of the theatre where the play was first presented, the name of the producer or director, and so forth. When the cast list relates to a specific performance, it should be embedded within a **<performance>** element (see section 10.1.3 ('Records of Performances') on p. 232), as in the following example:

```
<performance>
  <p>The first performance in Great Britain of <title>Waiting for
Godot</title> was given at the Arts Theatre, London, on
<date>3rd August 1955</>. It was directed by <name>Peter Hall</name>,
and the d&eacute;cor was by <name>Peter Snow</name>. The cast
was as follows:</p>
<castList>
   <castItem>Estragon: Peter Woodthorpe
   <castItem>Vladimir: Paul Daneman
   ...
  </castList>
</performance>
```

In this example, the **<castItem>** elements have no substructure. If desired, however, their components may be more finely distinguished using the elements **<role>**, **<roleDesc>**, and **<actor>**. For example, the first cast item above might be encoded as follows:

```
<castItem>
  <role id=VLAD>Vladimir</role>:
  <actor>Paul Daneman</actor>
</castItem>
```

The global id attribute may be used to specify a unique identifier for the <role> element, where it is desired to link speeches within the text explicitly to the role, using the who attribute, as further discussed in section 10.2.2 ('Speeches and Speakers') on p. 237 below.

The occasionally lengthy descriptions of a role sometimes found in written play scripts may be marked using the <roleDesc> element, as in the following example:

```
<castItem>
<role>Tom Thumb the Great</role>
<roleDesc>a little hero with a great soul, something
 violent in his temper, which is a little abated
 by his love for Huncamunca</roleDesc>
<actor>Young Verhuyk</actor>
</castItem>
```

For non-speaking or un-named roles, a <castItem> may contain a <roleDesc> without an accompanying <role>, for example

```
<castItem>
 <roleDesc>Costermonger</roleDesc>
</castItem>
```

When a list of such minor roles is given together, the type attribute of the <castItem> should indicate that it contains more than one role. The encoder may or may not elect to encode each separate constituent within such a composite <castItem>. Thus, either of the following is acceptable:

```
<castItem type=list>Constables, Drawer, Turnkey,  etc.</castItem>

<castItem type=list>
   <roleDesc>Constables,</roleDesc>
   <roleDesc>Drawer,</roleDesc>
   <roleDesc>Turnkey,</roleDesc>
   etc.
 </castItem>
```

A group of cast items forming a distinct subdivision of a cast list may be marked as such by using the special purpose <castGroup> element. The rend attribute should be used to indicate whether this grouping is indicated in the text by layout alone (i.e. the use of whitespace), by long braces or by some other means. A <castGroup> consists of an optional heading (represented as usual by a <head> element) followed by a series of <castItem> elements.

```
<castGroup rend=braced><head>friends of Mathias</head>
   <castItem>
       <role>Walter</role><actor>Mr Frank Hall</actor>
   </castItem>
   <castItem>
      <role>Hans</role><actor>Mr F.W. Irish</actor>
   </castItem>
</castGroup>
```

The following example demonstrates the use of the <castGroup> element to structure the whole of a <castList>, reflecting the way it is presented on the page:

```
<castList>
<castGroup>
  <head rend=braced>Mendicants</head>
  <castItem><role>Aafaa</>   <actor>Femi Johnson</></>
  <castItem><role>Blindman</><actor>Femi Osofisan</></>
  <castItem><role>Goyi</>    <actor>Wale Ogunyemi</></>
  <castItem><role>Cripple</> <actor>Tunji Oyelana</></>
</castGroup>
```

```
<castItem><role>Si Bero</>
          <roleDesc>Sister to Dr Bero</>
          <actor>Deolo Adedoyin</></>
<castGroup>
  <head rend=braced>Two old women</head>
  <castItem><role>Iya Agba</><actor>Nguba Agolia</></>
  <castItem><role>Iya Mate</><actor>Bopo George</></>
</castGroup>
<castItem><role>Dr Bero</>
          <roleDesc>Specialist</>
          <actor>Nat Okoro</></>
<castItem><role>Priest</>
          <actor>Gbenga Sonuga</></>
<castItem><role>The old man</>
          <roleDesc>Bero's father</>
          <actor>Dapo Adelugba</></>
</castlist>
```

The <castList> element and its components have the following formal definitions:

```
<!-- 10.1.4:  The cast list                                    -->
<!ELEMENT castList      - -  ((%m.divtop)*, (%component)*,
                             (castItem | castGroup)+,
                             (%component)*)                     >
<!ATTLIST castList           %a.global;                         >
<!ELEMENT castGroup     - -  (head?, (castItem | castGroup)+,
                             trailer?)                          >
<!ATTLIST castGroup          %a.global;                         >
<!ELEMENT castItem      - O  (role | roleDesc | actor |
                             (%phrase.seq))*                    >
<!ATTLIST castItem           %a.global;
          type               (role | list)       role          >
<!ELEMENT role          - O  (%phrase.seq)                      >
<!ATTLIST role               %a.global;                         >
<!ELEMENT roleDesc      - -  (%phrase.seq)                      >
<!ATTLIST roleDesc           %a.global;                         >
<!ELEMENT actor         - O  (%phrase.seq)                      >
<!ATTLIST actor              %a.global;                         >
<!-- This fragment is used in sec. 10.1                       -->
```

10.2 The Body of a Performance Text

The body of a performance text may be divided into structural units, variously called acts, scenes, stasima, entr'actes, etc. All such formal divisions should be encoded using an appropriate text-division element (, <div1>, <div2>, etc.), as further

discussed in section 10.2.1 ('Major Structural Divisions') on p. 236. Whether divided up into such units or not, all performance texts consist of sequences of speeches (see 10.2.2 ('Speeches and Speakers') on p. 237) and stage directions (see 10.2.3 ('Stage Directions') on p. 238). Speeches will generally consist of a sequence of *chunk*-level items: paragraphs, verse lines, stanzas, or (in case of uncertainty as to whether something is verse or prose) <seg> elements (see 10.2.4 ('Speech Contents') on p. 241).

The boundaries of formal units such as verse lines or paragraphs do not always coincide with speech boundaries. Units such as songs may be discontinuous or shared among several speakers. As described below in section 10.2.5 ('Embedded Structures') on p. 242, such fragmentation may be encoded in a relatively simple fashion using the linkage and aggregation mechanisms defined in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331.

10.2.1 Major Structural Divisions

Large divisions in drama such as acts, scenes, stasima, or entr'actes are indicated by numbered or unnumbered elements, as described in section 7.1 ('Divisions of the Body') on p. 185. The type and n attributes may be used to define the type of division being marked, and to provide a name or number for it, as in the following example:

```
<body>
  <div1 type=scene n=1>
    <head>Night&mdash;Faust's Study (i)</head>
    <!-- ... -->
  </div1>
  <div1 type=scene n=2>
    <head>Outside the City Gate</head>
    <!-- ... -->
  </div1>
</body>
```

Where the largest divisions of a performance text are themselves subdivided, most obviously in the case of plays traditionally divided into acts and scenes, further nested text-division elements may be used, as in this example:

```
<body>
  <div1 type=act n=1>
     <head>Act One</head>
     <div2 type=scene n='1'>
       <stage>Pa Ubu, Ma Ubu</stage>
       <sp><speaker>Pa Ubu</speaker><p>Pschitt!</p>
       <!-- ... -->
     </div2>
     <div2 type=scene n=2>
        <stage>A room in Pa Ubu's house, where a magnificent
        collation is set out</stage>
        <!-- ... -->
     </div2>
     <!-- ... -->
  </div1>
  <div1 type=act n=2>
    <head>Act Two</head>
    <div2 type=scene n=1>
      <head>Scene One</head>
      <!-- ... -->
    </div2>
    <div2 type=scene n=2>
      <head>Scene Two</head>
      <!-- ... -->
    </div2>
 </div1>
 <!-- ... -->
 </body>
```

In the example above, the <div2> element has been used to represent the "French scene" convention, (where the entrance of each new set of characters is marked as a distinct unit in the text) and the <div1> element to represent the acts into which the play is divided. The elements chosen are determined only by the hierarchic position of these units in the text as a whole. If the text had no acts, but only scenes, then the scenes might be represented by <div1> elements. Equally, if a play is divided only into "acts", with no smaller subdivisions, then the <div1> element might be used to represent acts. The type should be used, as above, where it is desired to make explicit the name associated with a particular category of subdivision.

As an alternative to the use of these "numbered" divisions, the encoder may represent all subdivisions with the same element, the "unnumbered" . The second act in the above example would then be represented as follows:

```
<div type=act n=2>
  <head>Act Two</head>
  <div type=scene n=1>
    <head>Scene One</head>
    <!-- ... -->
  </div>
  <div type=scene n=2>
    <head>Scene Two</head>
    <!-- ... -->
  </div>
</div>
```

For further discussion of the use of numbered and unnumbered divisions, see section 7.1 ('Divisions of the Body') on p. 185.

10.2.2 Speeches and Speakers

The following elements are used to identify speeches and speakers in a performance text:

<sp> An individual speech in a performance text, or a passage presented as such in a prose or verse text. Attributes include:

who identifies the speaker of the part by supplying an ID.

<speaker> A specialized form of heading or label, giving the name of one or more speakers in a dramatic text or fragment.

As noted above, the structure of many performance texts may be analysed as multiply hierarchic: a scene of a verse play, for example, may be divided into speeches and, at the same time, into verse lines. The end of a line may or may not coincide with the end of a speech, and vice versa. Other structures, such as songs, may be discontinuous or split up over several speeches. For some purposes it will be appropriate to regard the verse-structure as the fundamental organizing principle of the text, and for others the speech structure; in some cases, the choice between the two may be arbitrary. The discussion in the remainder of this chapter assumes that it is the speech-based hierarchy which most prominently determines the structure of performance texts, but the same mechanisms could be employed to encode a view of a performance text in which individual speeches were entirely subordinate to the formal units of prose and verse. SGML's CONCUR feature also allows for two (or more) such conflicting hierarchic views to be presented in a single document. For more detailed discussion and examples of alternative treatments of this fundamental issue, refer to chapter 31 ('Multiple Hierarchies') on p. 633.

The who attribute and the <speaker> element are both used to indicate the speaker or speakers of a speech, but in rather different ways. The <speaker> element is used to encode the word or phrase actually used within the source text to indicate the speaker: it may contain any string or prefix, and may be thought of as a highly specialized form of stage direction. The value of the who attribute however is a unique code, probably made up by the transcriber, which will unambiguously identify the character to whom the speech is assigned. To enforce this uniqueness, the base tag set for drama defines the value of this attribute as IDREFS. This means that the codes included in it must correspond with codes which are specified elsewhere in the document as identifiers for particular elements, typically the <role> element in the cast list where the character is named or described, as discussed in 10.1 ('Front and Back Matter ') on p. 228 above.

```
<!-- in the front matter ... -->
<castList>
<!-- ... -->
<castItem><role id=M2>Menaechmus</></>
<castItem><role id=Pen>Peniculus</></>
<!-- ... -->
</castList>
<!-- ... -->

<!-- in the text ... -->
```

```
<sp who=M2><speaker>Menaechmus</speaker>
  <l>Responde, adulescens, quaeso, quid nomen tibist?</l></sp>
<sp who=Pen><speaker>Peniculus</speaker>
  <l>Etiam derides, quasi nomen non noveris?</l></sp>
<sp who=M2><speaker>Menaechmus</speaker>
  <l>Non edepol ego te, quot sciam, umquam ante hunc diem</l>
  <l>Vidi neque novi; ...</l></sp>
```

If present, a <speaker> element may only appear as the first part of a <sp> element. The distinction between the <speaker> element and the who attribute makes it possible to encode uniformly characters whose names are not indicated in a uniform fashion throughout the play, or characters who appear in disguise, as in the following examples:

```
<castList>
  <!-- ... -->.
  <castItem><role id=HH>Henry Higgins</role></castItem>
  <!-- ... -->
</castList>
<!-- ... -->

<!-- in the text ... -->

<sp who="HH">
<speaker>The Notetaker</speaker>
<!-- ... -->
</sp>
```

If the speaker attributions are completely regular (and may thus be reconstructed mechanically from the values given for the who attribute), or are of no interest for the encoder of the text (as might be the case with editorially supplied attributions in older texts), then the <speaker> element need not be used; the former example above then might look like this:

```
<!-- in the front matter ... -->
<castList>
<castItem><role id=M2>Menaechmus</></>
<castItem><role id=Pen>Peniculus</></>
<!-- ... -->
</castList>
<!-- ... -->

<!-- in the text ... -->
<sp who=M2 ><l>Responde, adulescens, quaeso, quid nomen tibist?</l></sp>
<sp who=Pen><l>Etiam derides, quasi nomen non noveris?</l></sp>
<sp who=M2 ><l>Non edepol ego te, quot sciam, umquam ante hunc diem</l>
            <l>Vidi neque novi; ...</l></sp>
```

More than one IDREF may be listed as value for the who attribute if the speech is spoken by more than one person, as in the following example:

```
<stage>Nano and Castrone sing</stage>
<sp who="NAN CAS">
<l>Fools, they are the only nation
<l>Worth men's envy or admiration
<!-- ... -->
```

The <sp> and <speaker> elements are both declared within the core tag set (see section 6.11 ('Passages of Verse or Drama') on p. 176).

10.2.3 Stage Directions

Both between and within the speeches of a written performance text, it is normal practice to include a wide variety of descriptive directions to indicate non-verbal action. The following elements are provided to represent these:

<stage> contains any kind of stage direction within a dramatic text or fragment. Attributes include:

type indicates the kind of stage direction. Suggested values include:

setting describes a setting.

entrance describes an entrance.

exit describes an exit.

business describes stage business.

novelistic is a narrative, motivating stage direction.

delivery describes how a character speaks.

modifier gives some detail about a character.

location describes a location.

mixed more than one of the above

<move> marks the actual entrance or exit of one or more characters on stage. Attributes include:

who identifies the character or characters performing the movement.

type characterizes the movement, for example as an entrance or exit. Suggested values include:

entrance character is entering the stage.

exit character is exiting the stage.

onstage character moves on stage

where specifies the direction of a stage movement. Sample values include:

L stage left

R stage right

C centre stage

perf identifies the performance or performances in which this movement occurred as specified.

A satisfactory typology of stage directions is difficult to define. Certain basic types such as "entrance", "exit", "setting", "delivery", are easily identified. But the list is not a closed one, and it is not uncommon to mix types within a single direction. No closed set of values for the **type** attribute is therefore proposed at the present time, though some suggested values are indicated in the list below, which also indicates the range of possibilities.

```
<stage type="setting">The throne descends.</stage>
<stage type="setting">Music</stage>
<stage type="entrance">Enter Husband as being thrown off
       his horse.</stage>
<stage type="exit">Exit pursued by a bear.</stage>
<stage type="business">He quickly takes the stone out.</stage>
<stage type="delivery">To Lussurioso.</stage>
<stage type="delivery">Aside.</stage>
<stage type="delivery">Not knowing what to say.</stage>
<stage type="costume">Disguised as Ansaldo.</stage>
<stage type="location">At a window.</stage>
<stage type="novelistic">Having had enough, and embarrassed
        for the family.</stage>
```

Where possible, the values used for the **type** attribute on **<stage>** elements should be defined within the **<tagUsage>** element of the TEI header (described in section 5.3.4 ('The Tagging Declaration') on p. 98). For example:

```
<tagUsage gi=stage>This element is used for all stage directions,
editorial or authorial.  The TYPE attribute on this element takes
one or more of the following values:
<list type=gloss>
<label>setting</>   <item>describes the set</>
<label>blocking</>  <item>describes movement across stage,
position, etc.</>
```

```
<label>business</>  <item>describes movement other than blocking</>
<label>delivery</>  <item>describes how the line is said</>
<label>motivation</><item>describes character's emotional state
or through line</>
</list>
```

The <stage> element may appear both between and within <sp> elements. It may contain a mixture of phrase level elements, possibly combined into paragraphs, as in the following example:

```
<div1 type=act n=1>
<stage type=setting><p>
Scene. &mdash A room furnished comfortably and
tastefully but not extravagantly ...
The floor is carpeted and a fire burns in the stove.
It is winter.
<p>A bell rings in the hall; shortly afterwards the
door is heard to open. Enter NORA humming a tune ...
</stage>
<sp><speaker>Nora</speaker>
<p>Hide the Christmas Tree carefully, Helen. Be sure the
children do not see it till this evening, when it is
dressed. <stage type=delivery>To the PORTER taking
out her purse</stage> How much?
</sp>
```

The <stage> element may also be used in non-theatrical texts, to mark sound effects or musical effects, etc., as further discussed in section 10.3 ('Other Types of Performance Text') on p. 246.

The <move> element is intended to help overcome the fact that the stage directions of a printed text may often not provide full information about either the intended or the actual movement of actors etc. on stage. It may be used to keep track of entrances and exits in detail, so as to know which characters are on stage at which time. Its attributes permit a relatively formal specification for movements of characters, using user-defined codes to identify the characters involved (the who attribute), the direction of the movement (type attribute), and optionally which part of the stage is involved (where attribute). For stage-historical purposes, a perf attribute is also provided; this allows the recording of different <move> elements as taken in different performances of the same text.

The <move> element should be located at the position in the text where the move is presumed to take place. This will often coincide with a stage direction, as in the following simple example:

```
<stage type="entrance">
  <move who=B type="enter">
  Enter Bellafront mad.
</stage>
```

The <move> element can however appear independently of a stage direction, as in the following example:

```
<sp>
<speaker>Gent.</speaker>
<p>Neither to you, nor any one; having no witness
to confirm my speech. <move who=LM type=enter where=C>
Lo you! here she comes. This is her very guise; and,
upon my life, fast asleep.</p>
</stage>
```

The <stage> element is defined by the core TEI tag set (see section 6.11 ('Passages of Verse or Drama') on p. 176). The <move> element is defined as follows:

```
<!-- 10.2.3:  Stage directions                              -->
<!-- Stage is defined as part of the core.                  -->

<!ELEMENT move          - O  EMPTY                          >
```

```
<!ATTLIST move              %a.global;
          who               IDREFS              #REQUIRED
          type              CDATA               #IMPLIED
          where             CDATA               #IMPLIED
          perf              IDREFS              #IMPLIED       >
<!-- This fragment is used in sec. 10                          -->
```

10.2.4 Speech Contents

The actual speeches of a dramatic text may be composed of running text, which must be formally organized into paragraphs, in the case of prose (see section 6.1 ('Paragraphs') on p. 120), verse lines or line groups in that of verse (see section 6.11 ('Passages of Verse or Drama') on p. 176), or <seg> elements, in case of doubt as to whether the material should be treated as verse or prose. The following elements, all of which are defined in the core, are available for marking units of prose or verse within speeches:

<p> marks paragraphs in prose.

<lb> marks the start of a new (typographic) line in some edition or version of a text. Attributes include:

- **n** indicates the number or other value associated with the line which follows the point of insertion of this <lb>.
- **ed** indicates the edition or version in which the line break is located at this point

<l> contains a single, possibly incomplete, line of verse. Attributes include:

- **part** specifies whether or not the line is metrically complete. Legal values are:
  - ***Y*** the line is metrically incomplete
  - ***N*** either the line is complete, or no claim is made as to its completeness
  - ***I*** the initial part of an incomplete line
  - ***M*** a medial part of an incomplete line
  - ***F*** the final part of an incomplete line

<lg> contains a group of verse lines functioning as a formal unit e.g. a stanza, refrain, verse paragraph, etc.

<seg> contains any arbitrary phrase-level unit of text (including other <seg> elements).

As members of the class *metrical*, the elements <l> and <lg> share the following attributes:

met contains a user-specified encoding for the conventional metrical structure of the element.

rhyme specifies the rhyme scheme applicable to a group of verse lines.

As a member of the class *divn*, the element <lg> also bears the following attributes:

part specifies whether or not the division is fragmented by some other structural element, for example a speech which is divided between two or more verse stanzas. Legal values are:

- ***Y*** the division is incomplete in some respect
- ***N*** either the division is complete, or no claim is made as to its completeness.
- ***I*** the initial part of an incomplete division
- ***M*** a medial part of an incomplete division
- ***F*** the final part of an incomplete division

type specifies a name conventionally used for this level of subdivision, e.g. "act", "volume", "book", "section", "canto", etc.

In many texts, prose and verse may be inextricably mingled; particularly in earlier printed texts, prose may be printed as verse or verse as prose, or it may be impossible to distinguish the two. In cases of doubt, an encoder may prefer to tag the dubious material consistently as verse, to tag it all as prose, to follow the typography of the source text, or to use the neutral <seg> element to contain the speech itself. When this question arises, the <tagUsage> element in the <encodingDesc> element of the header should be used to record explicitly what policy has been adopted.

The part attribute of the <l> and <lg> elements provides one simple way of indicating where the boundaries of a speech and of a verse line or line group do not coincide. The encoder may simply indicate that a line or line group is incomplete by specifying the value Y or N, as in the following example:

```
<sp who=VOLP>
<!-- ... -->
<l part=Y>Now, now my clients
<l>Begin their visitation! Vulture, kite,
<l>Raven, and gorcrow, all my birds of prey,
<l>That think me turning carcase, now they come:
<l part=Y>I am not for 'em yet.
<stage>Re-enter MOSCA with the gown, etc.</stage>
<l part=Y>How now? the news?
</sp>
```

Alternatively, where the fragments of the line or line group are consecutive in the text (though possibly interrupted by stage directions), the values `i` (initial), `m` (medial) and `f` (final) may be used:

```
<sp who='Barnardo'> <!-- ... -->
 <l>If you do meet Horatio and Marcellus,</l>
 <l>The rivals of my watch, bid them make haste.</l></sp>
<stage>Enter Horatio and Marcellus.</>
<sp who='Francisco'>
 <l>I think I hear them. &mdash; Stand! Who's there?</l>
<sp who='Horatio'>
 <l part=i>Friends to this ground.</l>
<sp who='Marcellus'>
 <l part=f>And liegemen to the Dane.</l>
<!-- ... -->
```

In dramatic texts, the <lg> or line group element is most often of use for the encoding of songs and other stanzaic material, as further discussed in the next section. Line groups may be fragmented across speakers in the same way as individual lines, and the same set of attributes is available to record this fact. In the following example, a <lg> element is used to represent one verse of a song, which is divided between several voices:

```
<stage type=head>Song &mdash Sir Joseph</stage>
<sp who=JOPO>
<lg part=I>
<l>I am the monarch of the sea,
<l>The ruler of the Queen's Navee.
<l>Whose praise Great Britain loudly chants.
</lg></sp>
<sp who=HE><speaker>Cousin Hebe</speaker>
<lg part=M>
<l>And we are his sisters and his cousins and his aunts!
</lg></sp>
<sp who=REL><speaker>Rel.</speaker>
<lg part=F>
<l>And we are his sisters and his cousins and his aunts!
</lg></sp>
```

These elements are all defined in the core, and are thus available to every TEI document without formality. A more detailed discussion of the encoding of verse is provided in chapter 9 ('Base Tag Set for Verse ') on p. 213.

10.2.5 Embedded Structures

Although primarily composed of speeches, performance texts often contain other structural units such as songs or strophes which are shared among different speakers. More generally, complex nested structures of plays within plays, interpolated masques, or interludes are far from uncommon. In more modern material, comparably complex structural devices such as flashback or nested playback are equally frequent. In all kinds of performance material, it may be necessary to indicate several actions which are happening simultaneously.

A number of different devices are available within the TEI scheme to support these complexities in the general case. Texts may be composite or self-nesting (see section 7.3 ('Groups of

Texts') on p. 195) and multiple hierarchies may be defined (see chapter 31 ('Multiple Hierarchies') on p. 633). The TEI encoding scheme provides a variety of linking mechanisms, which may be used to indicate temporal alignment and aggregation of fragmented structures. In this section we provide a few specific examples of the application of these techniques to performance texts:

- the use of embedded <text> elements
- the use of the part attribute on fragmentary <lg> elements
- the use of the next and prev attributes on fragments of embedded structures to join them into a larger whole
- the use of the <join> element to define a "virtual element" composed of the fragments indicated

Full information and descriptions are provided in other chapters of this document, as indicated in the individual discussions.

When a song appears, in its entirety, within a single speech, it may be treated as an extended quotation or as an embedded <text> element, or both, according to the preference of the encoder. In the following example, an embedded song is treated as a self-standing text:

```
<sp><speaker>Kelly</> <stage>(calmly).</>
  <p>Aha, so you've bad minds along with th' love of gain.
     You thry to pin on others th' dirty decorations that
     may be hangin' on your own coats.
  <stage>(He points, one after the other at Conroy, Bull,
     and Flagonson.  Lilting):</>
  <q><text><body>
     <l>Who were you with last night?</l>
     <l>Who were you with last night?</l>
     <l>Will you tell your missus when you go home</l>
     <l>Who you were with last night?</l>
  </body></text></q></p></sp>
<sp><speaker>Flagonson</> <stage>(in anguished indignation).</>
  <p>This is more than a hurt to us:  this hits at the
     decency of the whole nation!</p></sp>
```

It might, however, also be treated simply as a quotation:

```
<sp><speaker>Kelly</> <stage>(calmly).</>
  <p>Aha, so you've bad minds along with th' love of gain.
     <!-- ... -->
  <stage>(He points, one after the other at Conroy, Bull,
     and Flagonson.  Lilting):</>
  <q><l>Who were you with last night?
     <l>Who were you with last night?
     <l>Will you tell your missus when you go home
     <l>Who you were with last night?
  </q>
</sp>
```

When an embedded structure extends across more than one <sp> element, each of its constituent parts must be regarded as a distinct fragment; the problem then facing the encoder is to reconstitute the interrupted whole in some way.

As already noted above, the part attribute may be used to indicate that an <l> element contains a partial, not a complete, verse line. The same attribute may be used on the <lg> element, to indicate that the line group is partial rather than complete, thus:

```
<sp><speaker>Kelly</>
    <stage>(wheeling quietly in his semi-dance, as he goes out):</>
    <lg part=i>
        <l>Goodbye to holy souls left here,</l>
        <l>Goodbye to man an' fairy;</l>
    </lg>
</sp>
<sp><speaker>Widda Machree</>
    <stage>(wheeling quietly in her semi-dance, as she goes out):</>
```

```
    <lg part=f>
        <l>Goodbye to all of Leicester Square,</l>
        <l>An' the long way to Tipperary.</l>
    </lg>
</sp>
```

When the fragments of a song are separated by other intervening dialogue, or even when not, they may be linked together with the next and prev attributes defined in section 14.7 ('Aggregation') on p. 369. For example, the line groups making up Ophelia's song might be encoded as follows:

```
<div1 type="act" n=4>
<!-- ... -->
<div2 type="scene" n=5>
<stage>Elsinore. A room in the Castle.</stage>
<!-- ... -->
<stage type="setting">Enter Ophelia, distracted.</stage>
<sp who="Oph"><speaker>Ophelia</speaker>
    <p>Where is the beauteous Majesty of Denmark?</p>
</sp>
<sp who="Qu"><speaker>Queen</speaker>
    <p>How now, Ophelia?</p>
</sp>
<sp who="Oph"><speaker>Ophelia</speaker>
    <stage>Singing</stage>
    <lg type=song ID="TL1" part=Y next=TL2 >
        <l>How should I your true-love know</l>
        <l>From another one?</l>
        <l>By his cockle hat and staff</l>
        <l>And his sandal shoon.</l>
    </lg>
</sp>
<sp who="Qu"><speaker>Queen</speaker>
    <p>Alas, sweet lady, what imports this song?</p>
</sp>
<sp who="Oph"><speaker>Ophelia</speaker>
    <p>Say you? Nay, pray you mark.</p>
    <stage>Sings</stage>
    <lg ID="TL2" part=Y prev=TL1>
        <l>He is dead and gone, lady,</l>
        <l>He is dead and gone;</l>
        <l>At his head a grass-green turf,</l>
        <l>At his heels a stone.</l>
    </lg>
    <p>O, ho!</p>
</sp>
<!-- ... -->
</div2>
</div1>
```

The next and prev attributes are discussed in section 14.7 ('Aggregation') on p. 369: they form part of the additional tag set for alignment and linking, and are therefore not automatically available to dramatic texts. To enable this tag set as well as the base tag set for drama, the document type declaration of a document might take the following form:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
  <!ENTITY % TEI.drama   'INCLUDE'>
  <!ENTITY % TEI.linking 'INCLUDE'>
]>
```

See chapter 3 ('Structure of the TEI Document Type Definition') on p. 35 for general discussion of the way in which TEI tag sets are enabled.

The fragments of Ophelia's song might also be linked together using the <join> mechanism

described in section 14.7 ('Aggregation') on p. 369. The <join> element is specifically intended to encode the fact that several discontiguous elements of the text together form one "virtual" element. Using this mechanism, the example might be encoded as follows:

```
<div1 type="act" n=4>
<!-- ... -->
<div2 type="scene" n=5>
<stage type="setting">Elsinore. A room in the Castle.</stage>
<!-- ... -->
<sp who="Qu"><speaker>Queen</speaker>
    <p>How now, Ophelia?</p>
</sp>
<sp who="Oph"><speaker>Ophelia</speaker>
    <stage type="delivery">Singing</stage>
    <lg type=song ID="TL1" part=Y>
        <l>How should I your true-love know</l>
        <l>From another one?</l>
        <l>By his cockle hat and staff</l>
        <l>And his sandal shoon.</l>
    </lg>
</sp>
<sp who="Qu">
<speaker>Queen</speaker>
    <p>Alas, sweet lady, what imports this song?</p>
</sp>
<sp who="Oph">
<speaker>Ophelia</speaker>
    <p>Say you? Nay, pray you mark.</p>
    <stage type="delivery">Sings</stage>
    <lg ID="TL2" part=Y >
        <l>He is dead and gone, lady,</l>
        <l>He is dead and gone;</l>
        <l>At his head a grass-green turf,</l>
        <l>At his heels a stone.</l>
    </lg>
<p>O, ho!</p>
<join parts='TL1 TL2' type=lg>
</sp>
<!-- ... -->
</div2>
</div1>
```

The location of the <join> element is not significant; here it has been placed shortly after the conclusion of the song, in order to have it close to the fragments it unifies.

Like the next and prev attributes, the <join> element requires the additional tag set for linking, which is selected as shown above.

10.2.6 Simultaneous Action

In printed or written versions of performance texts, a variety of techniques may be used to indicate the temporal alignment of speeches or actions. Speeches may be printed vertically aligned on the page, or braced together; stage directions (e.g. "Speaking at the same time") are also often used. In operatic or musical works in particular, the need to indicate timing and alignment of individual parts of a song may lead to very complex layout.

One simple method of indicating the temporal alignment of speeches or actions is to use the corresp attribute discussed in section 14.4 ('Correspondence and Alignment') on p. 358, as in the following example:

```
<sp who="M"><speaker>Mangan</>
  <stage type="delivery">wildly</stage>
  <p>Look here: I'm going to take off all my clothes.</p>
```

```
  <stage type="action">he begins tearing off his coat.</stage>
</sp>
<sp who="LU" id=S1><speaker>Lady Utterword</speaker>
  <p>Mr Mangan!</p>
</sp>
<sp who="CS" id=S2><speaker>Captain Shotover</speaker>
  <p>Whats that?</p>
</sp>
<sp who="H" id=S3><speaker>Hector</speaker>
  <p>Ha! ha! Do. Do.</p>
</sp>
<sp who="E" id=S4><speaker>Ellie</speaker>
  <p>Please dont.</p>
</sp>
<stage id=D1 type="delivery" corresp="S1 S2 S3 S4">
  in consternation</stage>

<sp who="MH"><speaker>Mrs. Hushabye</speaker>
  <stage type="action">catching his arm and stopping him</stage>
  <p>Alfred: for shame! Are you mad?</p>
</sp>
```

In the original, the stage direction "in consternation" is printed opposite a brace grouping all four speeches, indicating that all four characters speak at once, and that the stage direction applies to all of them. In the example, the <stage> element has been moved to an arbitrary place, and the four speeches with which it is to be associated are specified by identifier as the value of the corresp attribute. This attribute, which is enabled by the linking tag set, provides the simplest way of indicating the temporal alignment of speeches or actions in a play.

More powerful and more precise mechanisms for temporal alignment are defined in chapter 11 ('Transcriptions of Speech') on p. 249. These would be appropriate for encodings the focus of which is on the actual performance of a text rather than its structure or formal properties. The tag set described in that chapter includes a large number of other detailed proposals for the encoding of such features as voice quality, prosody, etc., which might be relevant to such a treatment of performance texts.

10.3 Other Types of Performance Text

Most of the elements and structures identified thus far are derived from traditional theatrical texts. Although other performance texts, such as screen plays or radio scripts, have not been discussed specifically, they can be encoded using the elements and structures listed above. Encoders may however find it convenient to use, as well, the additional specialized elements discussed in this section. For scripts containing very detailed technical information, the <tech> element discussed in section 10.3.1 ('Technical Information') on p. 248 may also be useful.

Like other texts, screen plays and television or radio scripts may be divided into text divisions marked with or <div1>, etc. Within units corresponding with the traditional "act" and "scene", further subdivisions or sequences may be identified, composed of individual "shots", each associated with a single camera angle and setting. Shots and sequences should be encoded using an appropriate text-division element (i.e., a <div3> element if numbered division elements are in use and the next largest unit is a <div2>, or a element if un-numbered divisions are in use) specifying sequence or shot as the value of the type attribute, as appropriate.

It is normal practice in screenplays and radio scripts to distinguish directions concerning camera angles, sound effects, etc., from other forms of stage direction. Such texts also generally include far more detailed specifications of what the audience actually sees: descriptions of actions and background, etc. Scripts derived from cinema and television productions may also include texts displayed as captions superimposed on the action. All of these may be encoded using the general purpose <stage> element discussed in section 10.2.3 ('Stage Directions') on p. 238, and

distinguished by means of its **type** attribute. Alternatively, or in addition, the following more specific elements may be used, where clear distinctions can be made:

<view> describes the visual context of some part of a screen play in terms of what the spectator sees, generally independent of any dialogue.

<camera> describes a particular camera angle or viewpoint in a screen play. Attributes include:

type characterizes the camera angle in some respect, e.g. as a close-up, medium shot, etc.

<caption> contains the text of a caption or other text displayed as part of a film script or screenplay.

<sound> describes a sound effect or musical sequence specified within a screen play or radio script. Attributes include:

type categorizes the sound in some respect, e.g. as music, special effect, etc.

discrete indicates whether the sound overlaps the surrounding speeches or interrupts them. Sample values include:

y the sound is heard between the surrounding speeches

n the sound overlaps the surrounding speeches

u unknown or inapplicable

Some examples of the use of these elements follow:

```
<camera>Angle on Olivia.</camera>
<view>Ryan's wife, standing nervously alone on the sidelines,
biting her lip. She's scared and she shows it.</view>
```

Where particular words or phrases within a direction are emphasized (by change of typeface or use of capital letters), an appropriate phrase-level element may be used to indicate the fact, as in the following examples, where certain words in the original are given in small capitals:

```
<view>George glances at the window--and freezes.
<camera>New angle--shock cut</camera> Out the window
the body of a dead man suddenly slams into
<hi>frame</hi>. He dangles grotesquely,
held up by his coat caught on a protruding bolt.
George gasps. The train <hi>whistle</hi> screams.
</view>
```

```
<view>Ext. TV control van&mdash;Early morning.
The <name>T.V. announcer</name> from the Ryan interview
stands near the Control Van, the lake in b.g.
</view>
<sp who="announcer">
<speaker>T.V. Announcer</speaker>
<p>Several years ago, Jack Ryan was a highly
successful hydroplane  racer....</p>
</sp>
```

All of these elements, like other stage directions, can appear both within and between speeches.

```
<sp>
<speaker>TV Announcer VO</speaker>
<p>Working with Ryan are his two coworkers&mdash;
Strut Bowman, the mechanical engineer&mdash;
<view><camera>Angle on Strut</camera>
standing in the tow boat, walkie-talkie in hand,
watching Ryan carefully.</view>
&mdash;and Roger Dalton, a rocket
systems analyst, and one of the scientists
from the Jet Propulsion Lab....</p></sp>
```

```
<sp><speaker>Benjy</speaker>
  <p>Now to business.</p></sp>
<sp><speaker>Ford and Zaphod</speaker>
  <p>To business.</p></sp>
```

```
  <sound>Glasses clink.</sound>
<sp><speaker>Benjy</speaker>
  <p>I beg your pardon?</p></sp>
<sp><speaker>Ford</speaker>
  <p>I'm sorry, I thought you were
  proposing a toast.</p></sp>

<camera>Zoom in to overlay showing some stock film
  of hansom cabs galloping past.</camera>
<caption>London, 1895.</caption>
<caption>The residence of Mr Oscar Wilde.</caption>
<sound>Suitably classy music starts.</sound>
<view>Mix through to Wilde's drawing room. A crowd of suitably
  dressed folk are engaged in typically brilliant conversation,
  laughing affectedly and drinking champagne.</view>
<sp who=TJ><speaker>Prince of Wales</speaker>
  <p>My congratulations, Wilde. Your latest play is a great success.
  </p></sp>
```

10.3.1 Technical Information

Traditional stage scripts may contain additional technical information about such production-related factors as lighting, "blocking" (that is, detailed notes on actors' movements) or props required at particular points. More technical information about intended production effects may also appear in published versions of screen plays or movie scripts. Where these are presented simply as marginal notes, they may be encoded using the general-purpose <note> element defined in section 6.8 ('Notes, Annotation, and Indexing') on p. 152. Alternatively, they may be formally distinguished from other stage directions by using the specialized <tech> element:

<tech> describes a special purpose stage direction that is not meant for the actors. Attributes include:

- **type** categorizes the technical stage direction. Suggested values include:
  - ***light*** a lighting cue.
  - ***sound*** a sound cue.
  - ***prop*** a prop cue.
  - ***block*** a blocking instruction
- **perf** identifies the performance or performances to which this technical direction applies.

Like stage directions, <tech> elements can appear anywhere within a speech or between speeches.

The elements discussed in the section are formally defined as follows:

```
<!-- 10.3.1:  Screen plays and other technical matters        -->
<!ELEMENT view          - O  (%specialPara)                     >
<!ATTLIST view               %a.global;                         >
<!ELEMENT camera        - -  (%paraContent)                     >
<!ATTLIST camera             %a.global;
          type               CDATA               #IMPLIED      >
<!ELEMENT sound         - O  (%paraContent)                     >
<!ATTLIST sound              %a.global;
          type               CDATA               #IMPLIED
          discrete           (y | n | u)         u             >
<!ELEMENT caption       - O  (%paraContent)                     >
<!ATTLIST caption            %a.global;                         >
<!ELEMENT tech          - O  (%paraContent)                     >
<!ATTLIST tech               %a.global;
          type               (light | sound | prop | block)
                                                 #IMPLIED
          perf               IDREFS              #IMPLIED      >
<!-- This fragment is used in sec. 10                          -->
```

Chapter 11

Transcriptions of Speech

The base tag set for transcriptions of spoken language described in this chapter is intended for use with a wide variety of transcribed spoken material. It should be stressed, however, that the present proposals are not intended to support unmodified every variety of research undertaken upon spoken material now or in the future; some discourse analysts, some phonologists, and doubtless others may wish to extend the scheme presented here to express more precisely the set of distinctions they wish to draw in their transcriptions. Speech regarded as a purely acoustic phenomenon may well require different methods from those outlined here, as may speech regarded solely as a process of social interaction.

This chapter begins with a discussion of some of the problems commonly encountered in transcribing spoken language (section 11.1 ('General Considerations and Overview') on p. 250). Section 11.2 ('Elements Unique to Spoken Texts') on p. 253 describes the basic structural elements of this tag set. Finally, section 11.3 ('Elements Defined Elsewhere') on p. 260 of this chapter reviews further problems specific to the encoding of spoken language, demonstrating how mechanisms and elements discussed elsewhere in these Guidelines may be applied to them.

The overall structure of a TEI spoken text is identical to that of any other TEI text: the <TEI.2> element for a spoken text contains a <teiHeader> element, followed by a <text> element. Even texts primarily composed of transcribed speech may also include conventional front and back matter, and may even be organized into divisions like printed texts. For simplicity's sake, therefore, the base tag set for spoken text uses the default text structure, as defined in chapter 7 ('Default Text Structure') on p. 183; this tag set is embedded automatically by the spoken base tag set.

To enable the base tag set for spoken texts, a parameter entity *TEI.spoken* must be declared within the document type declaration subset, the value of which is INCLUDE, as further described in section 3.3 ('Invocation of the TEI DTD') on p. 39. A document using this base tag set and no additional tag sets will thus begin as follows:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
     <!ENTITY % TEI.spoken 'INCLUDE' >
]>
```

This declaration makes available all of the elements and attributes discussed in the present chapter, in addition to the core elements described in chapter 6 ('Elements Available in All TEI Documents') on p. 119. If other elements are needed (in particular, those needed for synchronization or segmentation), additional tag sets may also be enabled in a similar way.

Two additional classes are defined by this tag set. Elements which appear only within transcribed speech constitute the *comp.spoken* element class. Elements with a specificable temporal duration constitute the *timed* element class. These classes are defined in the file *teispok2.ent* using the following parameter entities:

```
<!-- 11:  Class declarations for Transcribed Speech          -->
<!-- Text Encoding Initiative: Guidelines for Electronic     -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.   -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy   -->
```

```
<!-- in any form is granted, provided this notice is          -->
<!-- included in all copies.                                  -->

<!-- These materials may not be altered; modifications to     -->
<!-- these DTDs should be performed as specified in the       -->
<!-- Guidelines in chapter "Modifying the TEI DTD."           -->

<!-- These materials subject to revision. Current versions    -->
<!-- are available from the Text Encoding Initiative.         -->
<!ENTITY % x.comp.spoken ''                                    >
<!ENTITY % m.comp.spoken '%x.comp.spoken event | kinesic |
          pause | shift | u | vocal | writing'                 >
<!ENTITY % a.timed '
          start              IDREF              #IMPLIED
          end                IDREF              #IMPLIED
          dur                CDATA              #IMPLIED'     >
<!ENTITY % mix.spoken '| %m.comp.spoken'                       >
```

The elements of the base tag set for transcribed speech are declared in the file *teispok2.dtd*, which is organized as follows:

```
<!-- 11:  Base tag set for Transcribed Speech                 -->
<!-- Text Encoding Initiative: Guidelines for Electronic      -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.    -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy    -->
<!-- in any form is granted, provided this notice is          -->
<!-- included in all copies.                                  -->

<!-- These materials may not be altered; modifications to     -->
<!-- these DTDs should be performed as specified in the       -->
<!-- Guidelines in chapter "Modifying the TEI DTD."           -->

<!-- These materials subject to revision. Current versions    -->
<!-- are available from the Text Encoding Initiative.         -->
<!-- ... declarations from section 11.2.7                     -->
<!--     (Components of Transcribed Speech)                   -->
<!--     go here ...                                          -->

<!-- The base tag set for transcriptions of speech uses the   -->
<!-- standard default text-structure elements, which are      -->
<!-- embedded here:                                           -->

<![ %TEI.singleBase [
<!ENTITY % TEI.structure.dtd system 'teistr2.dtd'               >
%TEI.structure.dtd;
]]&nil;>
```

11.1 General Considerations and Overview

There is great variation in the ways different researchers have chosen to represent speech using the written medium.[1] This reflects the special difficulties which apply to the encoding or *transcription* of speech. Speech varies according to a large number of dimensions, many of which have no counterpart in writing (for example, tempo, loudness, pitch, etc.). The audibility of

[1] For a discussion of several of these see J. A. Edwards and M. D. Lampert, eds., *Talking Language: Transcription and Coding of Spoken Discourse* (Hillsdale, N.J.: Lawrence Erlbaum Associates, 1993); Stig Johansson, *Encoding a Corpus in Machine-Readable Form*, in *Computational Approaches to the Lexicon: An Overview*, ed. B. T. S. Atkins et al. (Oxford: Oxford University Press, forthcoming); and Stig Johansson et al. *Working Paper on Spoken Texts*, document TEI AI2 W1, 1991.

speech recorded in natural communication situations is often less than perfect, affecting the accuracy of the transcription. Spoken material may be transcribed in the course of linguistic, acoustic, anthropological, psychological, ethnographic, journalistic, or many other types of research. Even in the same field, the interests and theoretical perspectives of different transcribers may lead them to prefer different levels of detail in the transcript and different styles of visual display. The production and comprehension of speech are intimately bound up with the situation in which speech occurs, far more so than is the case for written texts. A speech transcript must therefore include some contextual features; determining which are relevant is not always simple. Moreover, the ethical problems in recording and making public what was produced in a private setting and intended for a limited audience are more frequently encountered in dealing with spoken texts than with written ones.

Speech also poses difficult structural problems. Unlike a written text, a speech event takes place in time. Its beginning and end may be hard to determine and its internal composition difficult to define. Most researchers agree that the utterances or *turns* of individual speakers form an important structural component in most kinds of speech, but these are rarely as well-behaved (in the structural sense) as paragraphs or other analogous units in written texts: speakers frequently interrupt each other, use gestures as well as words, leave remarks unfinished and so on. Speech itself, though it may be represented as words, frequently contains items such as vocalized pauses which, although only semi-lexical, have immense importance in the analysis of spoken text. Even non-vocal elements such as gestures may be regarded as forming a component of spoken text for some analytic purposes. Below the level of the individual utterance, speech may be segmented into units defined by phonological, prosodic, or syntactic phenomena; no clear agreement exists, however, even as to appropriate names for such segments.

Spoken texts transcribed according to the guidelines presented here are organized as follows. As noted above, speech is regarded as being composed of arbitrary high-level units called *texts*. A spoken <text> might typically be a conversation between a small number of people, a lecture, a broadcast TV item, or a similar event. Each such unit has associated with it a <teiHeader> providing detailed contextual information such as the source of the transcript, the identity of the participants, whether the speech is scripted or spontaneous, the physical and social setting in which the discourse takes place and a range of other aspects. For details of the header in general, refer to chapter 5 (‘The TEI Header’) on p. 77; for details of additional elements for the documentation of participant and contextual information, see section 23.2 (‘Contextual Information’) on p. 540.

Defining the bounds of a spoken text is frequently a matter of arbitrary convention or convenience. In public or semi-public contexts, a text may be regarded as synonymous with, for example, a *lecture*, a *broadcast item*, a *meeting*, etc. In informal or private contexts, a text may be simply a conversation involving a specific group of participants. Alternatively, researchers may elect to define spoken texts solely in terms of their duration in time or length in words. By default, these Guidelines assume of a text only that:

- it is internally cohesive,
- it is describable by a single header, and
- it represents a single stretch of time with no significant discontinuities.

Deviation from these assumptions may be specified (for example, the org attribute on the <text> element may take the value compos to specify that the components of the text are discrete) but is not recommended.

Within a <text> it may be necessary to identify subdivisions of various kinds, if only for convenience of handling. The neutral element discussed in section 7.1 (‘Divisions of the Body’) on p. 185 is recommended for this purpose. It may be found useful also for representing subdivisions relating to discourse structure, speech act theory, transactional analysis, etc., provided that these divisions are hierarchically well-behaved. Where they are not, as is often the case, the mechanisms discussed in chapters 14 (‘Linking, Segmentation, and Alignment’) on p. 331 and 31 (‘Multiple Hierarchies’) on p. 633 may be used.

A spoken text may contain any of the following components:

- utterances
- pauses

- vocalized but non-lexical phenomena such as coughs
- kinesic (non-verbal, non-lexical) phenomena such as gestures
- entirely non-linguistic events occurring during and possibly influencing the course of speech
- writing, regarded as a special class of event in that it can be transcribed, for example captions or overheads displayed during a lecture
- shifts or changes in vocal quality

Elements to represent all of these features of spoken language are discussed in section 11.2 ('Elements Unique to Spoken Texts') on p. 253 below.

An utterance (tagged <u>) may contain lexical items interspersed with pauses and non-lexical vocal sounds; during an utterance, non-linguistic events may occur and written materials may be presented. The <u> element can thus contain any of the other elements listed, interspersed with a transcription of the lexical items of the utterance; the other elements may all appear between utterances or next to each other, but except for <writing> they do not contain any other elements nor any data.

11.1.1 Divisions

A spoken text itself may be without substructure, that is, it may consist simply of units such as utterances or pauses, not grouped together in any way, or it may be subdivided into one or more divisions as described in this section.

If the notion of what constitutes a "text" in spoken discourse is inevitably rather an arbitrary one, the notion of formal subdivisions within such a "text" is even more debatable. Nevertheless, such divisions may be useful for such types of discourse as debates, broadcasts, etc., where structural subdivisions can easily be identified, or more generally wherever it is desired to aggregate utterances or other parts of a transcript into units smaller than a complete "text". Examples might include "conversations" or "discourse fragments", or more narrowly, "that part of the conversation where topic x was discussed", provided only that the set of all such divisions is coextensive with the text.

Each such division of a spoken text should be represented by the numbered or un-numbered elements defined in chapter 7 ('Default Text Structure') on p. 183. For some detailed kinds of analysis a hierarchy of such divisions may be found useful; nested elements may be used for this purpose, as in the example below.

The element is a member of the *divn* class of structural elements, and therefore has the following attributes in common with other members of the class:

type specifies a name conventionally used for this level of subdivision, e.g. "act", "volume", "book", "section", "canto", etc.

org specifies how the content of the division is organized. Legal values are:

composite composite content: i.e. no claim is made about the sequence in which the immediate contents of this division are to be processed, or their inter-relationships.

uniform uniform content: i.e. the immediate contents of this element are regarded as forming a logical unit, to be processed in sequence.

sample indicates whether this division is a sample of the original source and if so, from which part. Legal values are:

initial division lacks material present at end in source.

medial division lacks material at start and end.

final division lacks material at start.

unknown position of sampled material within original unknown.

complete division is not a sample.

part specifies whether or not the division is fragmented by some other structural element, for example a speech which is divided between two or more verse stanzas. Legal values are:

Y the division is incomplete in some respect

N either the division is complete, or no claim is made as to its completeness.

I the initial part of an incomplete division

M a medial part of an incomplete division

F the final part of an incomplete division

The type attribute may be used to characterize divisions in any way that is convenient; no specific recommendations are made in these Guidelines. For example, a collection made up of transcribed "sound bites" taken from speeches given by a politician on different occasions, might encode each extract as a distinct , nested within a single composite as follows:

```
<div type='soundbites' org='composite'>

   <div part=medial complete=no>
   <!-- ... -->
   </div>
   <div part=medial complete=no>
   <!-- ... -->
   </div>
   <div part=initial complete=no>
   <!-- ... -->
   </div>
</div>
```

As a member of the class *declaring*, the element may also carry a decls attribute, for use where the divisions of a text do not all share the same set of the contextual declarations specified in the TEI header. (See further section 23.3 ('Associating Contextual Information with a Text') on p. 550).

11.2 Elements Unique to Spoken Texts

The following elements characterize spoken texts, transcribed according to these Guidelines:

<u> a stretch of speech usually preceded and followed by silence or by a change of speaker.

<pause> a pause either between or within utterances.

<vocal> any vocalized but not necessarily lexical phenomenon, for example voiced pauses, non-lexical backchannels, etc.

<kinesic> any communicative phenomenon, not necessarily vocalized, for example a gesture, frown, etc.

<event> any phenomenon or occurrence, not necessarily vocalized or communicative, for example incidental noises or other events affecting communication.

<writing> a passage of written text revealed to participants in the course of a spoken text.

<shift> marks the point at which some paralinguistic feature of a series of utterances by any one speaker changes.

Each of these is further discussed and specified below in sections 11.2.1 ('Utterances') on p. 254 to 11.2.4 ('Writing') on p. 257.

We can show the relationship between four of these constituents of speech using the features *eventive*, *communicative*, *anthropophonic* (for sounds produced by the human vocal apparatus), and *lexical*:

	eventive	communicative	anthropophonic	lexical
event	+	-	-	-
kinesic	+	+	-	-
vocal	+	+	+	-
utterance	+	+	+	+

The differences are not always clear-cut. Among *events* might be included actions like slamming the door, which can certainly be communicative. *Vocals* include coughing and sneezing, which are usually involuntary noises. Equally, the distinction between utterances and vocals is not always clear, although for many analytic purposes it will be convenient to regard them as distinct. Individual scholars may differ in the way borderlines are drawn and should declare their definitions in the <editorialDecl> element of the header (see 5.3.3 ('The Editorial Practices Declaration') on p. 96).

The following short extract exemplifies several of these elements. It is recoded from a text originally transcribed in the CHILDES format.[2] Each utterance is encoded using a <u> element (see section 11.2.1 ('Utterances') on p. 254). Pauses marked by the transcriber are indicated using the <pause> element (see section 11.2.2 ('Pause') on p. 255). Non-verbal vocal effects such as the child's meowing are indicated either with orthographic transcriptions or with the <vocal> element, and entirely non-linguistic but significant events such as the sound of the toy cat are represented by the <event> elements (see section 11.2.3 ('Vocal, Kinesic, Event') on p. 256).

```
<u who=MAR>you never <pause> take this cat for show&sp;and&sp;tell
  <pause> meow meow</u>
<u who=ROS>yeah well I dont want to</u>
<event desc=
  'toy cat has bell in tail which continues to make a tinkling sound'>
<vocal who=MAR desc='meows'>
<u who=ROS>because it is so old</u>
<u who=MAR>how <reg orig="bout">about</reg> your&stress; cat <pause>
   yours is  new &stress;
  <kinesic desc='shows Father the cat'></u>
<u who=FAT trans=pause>thats <pause> darling</u>
<u who=MAR><s>no mine&stress; isnt old</s>
          <s>mine is just um a little dirty</s></u>
```

This example also uses some elements common to all TEI texts, notably the <reg> tag for editorial regularization. Special purpose entity references have been used to indicate non-separating spaces (&sp) and unusually stressed syllables (&stress); an alternative to the latter might have been to use the core <emph> element. The <s> element has also been used to segment the last utterance. Further discussion of all of these options is provided in section 11.3 ('Elements Defined Elsewhere') on p. 260.

Contextual information is of particular importance in spoken texts, and is usually provided by the TEI header of a text. In general, all of the information in a header is understood to be relevant to the whole of the associated text. The elements <u> and <writing> are however members of the *declaring* class, and may therefore specify a different context from that of the surrounding elements within a given division or text by means of the decls attribute (see further section 23.3 ('Associating Contextual Information with a Text') on p. 550).

11.2.1 Utterances

Each distinct *utterance* in a spoken text is represented by a <u> element, described as follows:

<u> a stretch of speech usually preceded and followed by silence or by a change of speaker. Attributes include:

- **who** supplies an identifier for the speaker or group of speakers. Its value is the identifier of a <participant> or <participantGrp> element in the TEI header.
- **trans** indicates the nature of the transition between this utterance and the previous one. Sample values include:
  - ***smooth*** this utterance begins without unusual pause or rapidity.
  - ***latching*** this utterance begins with a markedly shorter pause than normal.
  - ***overlap*** this utterance begins before the previous one has finished.
  - ***pause*** this utterance begins after a noticeable pause.

Use of the who attribute to associate the utterance with a particular speaker is recommended but not required. Its use implies as a further requirement that all speakers be identified by a <participant> or <participantGrp> element in the TEI header (see section 23.2.2 ('The Participants Description') on p. 545). Where utterances cannot be attributed with confidence to any particular participant or group of participants, the encoder may choose to define "participants" such as *all* or *various*. For example:

[2] The original is a conversation between two children and their parents, recorded in 1987, and discussed in Brian MacWhinney, *CHAT Manual* ([Pittsburgh]: Dept of Psychology, Carnegie-Mellon University, 1988), pp. 87ff.

```
<u who=A>     <!-- utterance by speaker A -->
<u who='A B'> <!-- utterance by speakers A and B -->
<u who='ALL'> <!-- utterance by speaker group ALL -->
```

The trans attribute is provided as a means of characterizing the transition from one utterance to the next at a simpler level of detail than that provided by the temporal alignment mechanism discussed in section 14.5 ('Synchronization') on p. 364. The value specified applies to the transition from the preceding utterance into the utterance bearing the attribute. For example:[3]

```
<u id=A1 who=A>Have you heard the</u>
<u id=B1 who=B trans=latching>the election results? yes</u>
<u id=A2 who=A trans=pause>it's a disaster</u>
<u id=B2 who=B trans=overlap>it's a miracle</u>
```

In this example, utterance B1 latches on to utterance A1, while there is a marked pause between B1 and A2. B2 and A2 overlap, but by an unspecified amount. For ways of providing a more precise indication of the degree of overlap, see section 11.3.2 ('Synchronization and Overlap') on p. 262.

An utterance may contain either running text, or text within which other basic structural elements are nested. Where such nesting occurs, the who attribute is considered to be inherited for the elements <pause>, <vocal>, <shift> and <kinesic>; that is, a pause or shift (etc.) within an utterance is regarded as being produced by that speaker only, while a pause between utterances applies to all speakers.

Occasionally, an utterance may contain other utterances, for example where there is a change in the script associated with it. This may occur when a speaker changes script in mid-utterance. For example:

```
<!-- breakfast table conversation ... -->
<u who=A>Listen to this
   <u who=A decls=S1>The government is confident, he
   said, that the current economic problems will be
   completely overcome by June</u>
what nonsense</u>
```

Here speaker A's own utterance contains a second nested utterance, which is read from a newspaper. The decls attribute on the nested utterance is used to indicate that its script is S1, rather than the default. Alternatively, the embedded utterance might be regarded as a new (non-nested) one. It might also be encoded using the <writing> element described in section 11.2.3 ('Vocal, Kinesic, Event') on p. 256 below, or the <event> element described in section 11.2.3 ('Vocal, Kinesic, Event') on p. 256, without transcribing the read material:

```
<u who=A>Listen to this <event desc='reads'>
what nonsense</u>
```

11.2.2 Pause

The <pause> empty element is used to indicate a perceived pause, either between or within utterances.

<pause> a pause either between or within utterances. Attributes include:

- **who** supplies an identifier for the person or group pausing. Its value is the identifier of a <participant> or <participant.grp> element in the TEI header.
- **type** categorizes the pause in some respect.

A pause contained by an utterance applies to the speaker of that utterance. A pause between utterances applies to all speakers. The type attribute may be used to categorize the pause, for example as short, medium or long; alternatively the attribute dur may be used to indicate its length more exactly, as in the following example:

[3] For the most part, the examples in this chapter use no sentence punctuation except to mark the rising intonation often found in interrogative statements; for further discussion, see section 11.3.3 ('Regularization of Word Forms') on p. 266.

```
<u>Okay <pause dur=200>U-m<pause dur=75>the s the scene
opens up <pause dur=50> with <pause dur=20> um <pause dur=145>
you see a tree okay? </u>
```

If detailed synchronization of pausing with other vocal phenomena is required, the alignment mechanism defined at section 14.5 ('Synchronization') on p. 364 and discussed informally below should be used. Note that the trans attribute mentioned in the previous section may also be used to characterize the degree of pausing between (but not within) utterances.

11.2.3 Vocal, Kinesic, Event

These three empty elements are used to indicate the presence of non-transcribed semi-lexical or non-lexical phenomena either between or within utterances.

<vocal> any vocalized but not necessarily lexical phenomenon, for example voiced pauses, non-lexical backchannels, etc. Attributes include:

- **who** supplies an identifier for the vocalist(s). Its value is the identifier of a <participant> or <participant.grp> element in the TEI header.
- **desc** supplies a conventional representation for the phenomenon.
- **iterated** indicates whether or not the phenomenon is repeated. Sample values include:
  - ***y*** the phenomenon is repeated.
  - ***n*** the phenomenon is atomic.
  - ***u*** unknown or unmarked.

<kinesic> any communicative phenomenon, not necessarily vocalized, for example a gesture, frown, etc. Attributes include:

- **who** supplies an identifier for the participant performing the gesture. Its value is the identifier of a <participant> or <participant.grp> element in the TEI header.
- **desc** supplies a conventional representation for the phenomenon.
- **iterated** indicates whether or not the phenomenon is repeated. Sample values include:
  - ***y*** the phenomenon is repeated.
  - ***n*** the phenomenon is atomic.
  - ***u*** unknown or unmarked.

<event> any phenomenon or occurrence, not necessarily vocalized or communicative, for example incidental noises or other events affecting communication. Attributes include:

- **who** supplies an identifier for the agent of the event described, if any. Its value is the identifier of a <participant> or <participant.grp> element in the TEI header.
- **desc** supplies a conventional representation for the phenomenon.
- **iterated** indicates whether or not the phenomenon is repeated. Sample values include:
  - ***y*** the phenomenon is repeated.
  - ***n*** the phenomenon is atomic.
  - ***u*** unknown or unmarked.

The who attribute should be used to specify the person or group responsible for a vocal, kinesic or event which is contained within an utterance, if this differs from that of the enclosing utterance. The attribute must be supplied for a vocal, kinesic or event which is not contained within an utterance.

The iterated attribute may be used to indicate that the vocal, kinesic or event is repeated, for example laughter as opposed to laugh. These should both be distinguished from laughing, where what is being encoded is a shift in voice quality. For this last case, the <shift> element discussed in section 11.2.6 ('Shifts') on p. 258 should be used.

The desc attribute may be used to supply a conventional representation for the phenomenon, for example:

non-lexical burp, click, cough, exhale, giggle, gulp, inhale, laugh, sneeze, sniff, snort, sob, swallow, throat, yawn

semi-lexical ah, aha, aw, eh, ehm, er, erm, hmm, huh, mm, mmhm, oh, ooh, oops, phew, tsk, uh, uh-huh, uh-uh, um, urgh, yup

Researchers may prefer to regard some semi-lexical phenomena as "words" within the bounds of the <u> element. See further the discussion at section 11.3.3 ('Regularization of Word Forms') on p. 266 below. As for all basic categories, the definition should be made clear in the <encodingDesc> element of the TEI header.

Some typical examples follow (recoded from J. Maxwell Atkinson and John Heritage, eds., *Transcript notation. Structures of social action: Studies in conversation analysis,* Cambridge University Press, 1984).

```
<u who=Jan>This is just delicious</u>
<event desc='telephone rings'>
<u who=Kim>I'll get it</u>

<u who=Tom>I used to <vocal desc=cough> smoke a lot</u>
<u who=Bob><vocal desc=sniff>He thinks he's tough</u>
<vocal who=Ann desc=snorts>
```

Note that Ann's snorting could equally well be encoded as follows:

```
<u who=Ann><vocal desc=snorts></u>
```

The extent to which encoding of events or kinesics is included in a transcription will depend entirely on the purpose for which the transcription was made. As elsewhere, this will depend on the particular research agenda and the extent to which their presence is felt to be significant for the interpretation of spoken interactions.

11.2.4 Writing

Written text may also be encountered when speech is transcribed, for example in a television broadcast or cinema performance, or where one participant shows written text to another. The <writing> element may be used to distinguish such written elements from the spoken text in which they are embedded.

<writing> a passage of written text revealed to participants in the course of a spoken text. Attributes include:

- **who** supplies an identifier for the participant who reveals or creates the writing, if any. Its value is the identifier of a <participant> or <participant.grp> element in the TEI header.
- **gradual** indicates whether the writing is revealed all at once or gradually. Sample values include:
  - ***y*** the writing is revealed gradually.
  - ***n*** the writing is revealed all at once.
  - ***u*** unknown or unmarked.
- **type** categorizes the kind of writing in some way, for example as a subtitle, noticeboard etc.

For example, if speaker A in the breakfast table conversation in section 11.2.1 ('Utterances') on p. 254 above had simply shown the newspaper passage to her interlocutor instead of reading it, the interaction might have been encoded as follows:

```
<u who=A>look at this
<writing who=A gradual=N type=newspaper><p>
  The government is confident, he said, that the
  current economic problems will be completely overcome
  by June</writing>
<u who=A>what nonsense!
```

11.2.5 Temporal Information

In addition to the global attributes n, id, and lang, utterances, vocals, pauses, kinesics, events and writing elements may all take a common set of attributes providing information about their position in time. For this reason, these elements are regarded as forming a *class*, referred to here as *timed*. The following attributes are common to all elements in this class:

start indicates the location within a temporal alignment at which this element begins.

end indicates the location within a temporal alignment at which this element ends.
dur indicates the length of this element in time, using either specific units or the units specified on the associated temporal alignment.

Note that if start and end point to <when> elements whose temporal distance from each other is specified in a timeline, then dur is ignored.

The <anchor> element (see 14.4 ('Correspondence and Alignment') on p. 358) may be used as an alternative means of aligning the start and end of timed elements, and is required when the temporal alignment involves points within an element.

11.2.6 Shifts

A common requirement in transcribing spoken language is to mark positions at which a variety of prosodic features change. Many paralinguistic features (pitch, prominence, loudness, etc.) characterize stretches of speech which are not co-extensive with utterances or any of the other units discussed so far. One simple method of encoding such units is simply to mark their boundaries. An empty element called <shift> is provided for this purpose.

<shift> marks the point at which some paralinguistic feature of a series of utterances by any one speaker changes. Attributes include:

feature a paralinguistic feature. Sample values include:

tempo speed of utterance.
loud loudness.
pitch pitch range.
tension tension or stress pattern.
rhythm rhythmic qualities.
voice voice quality.

new specifies the new state of the paralinguistic feature specified.

A <shift> element may appear within an utterance or a segment to mark a significant change in the particular feature defined by its attributes, which is then understood to apply to all subsequent utterances for the same speaker, unless changed by a new shift for the same feature in the same speaker. Intervening utterances by other speakers do not normally carry the same feature.

For example:

```
<u who=LB><shift feature=loud new=f>Elizabeth</u>
<u who=EB>Yes</u>
<u who=LB><shift>Come and try this <pause>
     <shift feature=loud new=ff>come on</u>
```

In this example, the word 'Elizabeth' is spoken loudly, the words 'Yes' and 'Come and try this' with normal volume, and the words 'come on' very loudly.

The values proposed here for the feature attribute are based on those used by the Survey of English Usage; [4] this list may be revised or supplemented using the methods outlined in section 29 ('Modifying the TEI DTD') on p. 619.

The new attribute specifies the new state of the feature following the shift. If no value is specified, it is implied that the feature concerned ceases to be remarkable at this point: the special value normal may be specified to have the same effect.

A list of suggested values for each of the features proposed follows:

- tempo

a allegro (fast)
aa very fast
acc accelerando (getting faster)
l lento (slow)
ll very slow

[4] For details see S. Boase, *London-Lund Corpus: Example Text and Transcription Guide* (London: Survey of English Usage, University College London, 1990).

rall rallentando (getting slower)

- loud (for loudness):

  f forte (loud)
  ff very loud
  cresc crescendo (getting louder)
  p piano (soft)
  pp very soft
  dimin diminuendo (getting softer)

- pitch (for pitch range):

  high high pitch-range
  low low pitch-range
  wide wide pitch-range
  narrow narrow pitch-range
  asc ascending
  desc descending
  monot monotonous
  scand scandent, each succeeding syllable higher than the last, generally ending in a falling tone

- tension:

  sl slurred
  lax lax, a little slurred
  ten tense
  pr very precise
  st staccato, every stressed syllable being doubly stressed
  leg legato, every syllable receiving more or less equal stress

- rhythm:

  rh beatable rhythm
  arrh arrhythmic, particularly halting
  spr spiky rising, with markedly higher unstressed syllables
  spf spiky falling, with markedly lower unstressed syllables
  glr glissando rising, like spiky rising but the unstressed syllables, usually several, also rise in pitch relative to each other
  glf glissando falling, like spiky falling but with the unstressed syllables also falling in pitch relative to each other

- voice (for voice quality):

  whisp whisper
  breath breathy
  husk husky
  creak creaky
  fals falsetto
  reson resonant
  giggle unvoiced laugh or giggle
  laugh voiced laugh
  trem tremulous
  sob sobbing
  yawn yawning
  sigh sighing

A full definition of the sense of the values provided for each feature should be provided in the encoding description section of the text header (see section 5.3 ('The Encoding Description') on p. 93).

11.2.7 Formal Definition

The components of the tag set for transcribed speech are formally defined as follows:

```
<!-- 11.2.7:  Components of Transcribed Speech                 -->
<!ELEMENT u             - -  ((%phrase | %m.comp.spoken)+)      >
<!ATTLIST u                  %a.global;
                             %a.timed;
          trans              (smooth | latching | overlap |
                             pause)              smooth
          who                IDREF               %INHERITED     >
<!ELEMENT pause         - O  EMPTY                              >
<!ATTLIST pause              %a.global;
                             %a.timed;
          type               CDATA               #IMPLIED
          who                IDREF               #IMPLIED       >
<!ELEMENT vocal         - O  EMPTY                              >
<!ATTLIST vocal              %a.global;
                             %a.timed;
          who                IDREF               %INHERITED
          iterated           (y | n | u)         n
          desc               CDATA               #IMPLIED       >
<!ELEMENT kinesic       - O  EMPTY                              >
<!ATTLIST kinesic            %a.global;
                             %a.timed;
          who                IDREF               %INHERITED
          iterated           (y | n | u)         n
          desc               CDATA               #IMPLIED       >
<!ELEMENT event         - O  EMPTY                              >
<!ATTLIST event              %a.global;
                             %a.timed;
          who                IDREF               %INHERITED
          iterated           (y | n | u)         n
          desc               CDATA               #IMPLIED       >
<!ELEMENT writing       - -  (%paraContent;)                    >
<!ATTLIST writing            %a.global;
          who                IDREF               %INHERITED
          type               CDATA               #IMPLIED
          script             IDREF               #IMPLIED
          gradual            (y | n | u)         #IMPLIED       >
<!ELEMENT shift         - O  EMPTY                              >
<!ATTLIST shift              %a.global;
          who                IDREF               #IMPLIED
          feature            (tempo | loud | pitch | tension |
                             rhythm | voice)     #REQUIRED
          new                CDATA               normal         >
<!-- This fragment is used in sec. 11                          -->
```

11.3 Elements Defined Elsewhere

This section describes the following features characteristic of spoken texts for which elements are defined elsewhere in these Guidelines:

- segmentation below the utterance level
- synchronization and overlap
- regularization of orthography

The elements discussed here are not provided by the base tag set for spoken texts. Some of them are included in the core tag set available to all TEI documents, but others are contained in the TEI additional tag sets for linking and for analysis respectively. To enable these tag sets, the appropriate parameter entities must be declared in the document type declaration subset, as described in section 3.3 ('Invocation of the TEI DTD') on p. 39. For example, if a transcript using the base tag set defined in this chapter additionally wishes to make use of the <timeLine>

element, then the following declarations would be necessary:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
     <!ENTITY % TEI.spoken   'INCLUDE' >
     <!ENTITY % TEI.linking  'INCLUDE'>
]>
```

If the complex segmentation elements defined in the additional tag set for analysis were also required, the following declarations would be needed:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
     <!ENTITY % TEI.spoken   'INCLUDE'>
     <!ENTITY % TEI.linking  'INCLUDE'>
     <!ENTITY % TEI.analysis 'INCLUDE'>
]>
```

11.3.1 Segmentation

For some analytic purposes it may be desirable to subdivide the divisions of a spoken text into units smaller than the individual utterance or turn. Segmentation may be performed for a number of different purposes and in terms of a variety of speech phenomena. Common examples include units defined both prosodically (by intonation, pausing, etc.) and syntactically (clauses, phrases, etc.) The term *macrosyntagm* has been used by a number of researchers to define units peculiar to speech transcripts. [5]

These Guidelines propose that such analyses be performed in terms of neutrally-named *segments*, represented by the <seg> element, which is discussed more fully in section 14.3 ('Segments and Anchors') on p. 355. This element may take a type attribute to specify the kind of segmentation applicable to a particular segment, if more than one is possible in a text. A full definition of the segmentation scheme or schemes used should be provided in the <segmentation> element of the <editorialDecl> element in the TEI header (see 5.3.3 ('The Editorial Practices Declaration') on p. 96).

In the first example below, an utterance has been segmented according to a notion of syntactic completeness not necessarily marked by the speech, although in this case a pause has been recorded between the two sentence-like units. In the second, the segments are defined prosodically (an entity reference &stress; has been used to mark the position immediately following the syllable bearing the primary accent or stress), and may be thought of as "tone units".

```
<u who=M1><seg>we went to the pub yesterday</seg>
    <pause><seg>there was no one there</seg>
</u>
<u who=F1><seg>although its an old ide&stress;a</seg>
    <seg>it hasnt been on the mar&stress;ket very long</seg>
</u>
```

In either case, the <segmentation> element in the header of the text should specify the principles adopted to define the segments marked in this way.

When utterances are segmented end-to-end in the same way as the s-units in written texts, the <s> element discussed in chapter 15 ('Simple Analytic Mechanisms') on p. 381 may be used, either as an alternative or in addition to the more general purpose <seg> element. The <s> element is available without formality in all texts, but does not allow segments to nest within each other.

Where segments of different kinds are to be distinguished within the same stretch of speech, the type attribute may be used, as in the following example. The example also shows the use of a user-specified extension to the TEI tag sets, for specifying paraphasia.

[5] The term was apparently first proposed by Bengt Loman and Nils Jørgensen, in *Manual for analys och beskrivning av makrosyntagmer* (Lund: Studentlitteratur, 1971), where it is defined as follows: "A text can be analysed as a sequence of segments which are internally connected by a network of syntactic relations and externally delimited by the absence of such relations with respect to neighbouring segments. Such a segment is a syntactic unit called a macrosyntagm" (trans. S. Johansson).

```
<u who=T1>
<seg type=C>I think </seg>
<seg type=C>this chap was writing </seg>
<seg type=C>and he <del type=repeated>said hello</del> said </seg>
<seg type=M>hello </seg>
<seg type=C>and he said </seg>
<seg type=C>I'm going to a <paraphasia>gate</paraphasia>
            at twenty past seven </seg>
<seg type=C>he said </seg>
<seg type=M>ok </seg>
<seg type=M>right away </seg>
<seg type=C>and so <omit desc=unclear extent='1'> on they went </seg>
<seg type=C>and they were <omit desc=unclear extent='3'>
            writing there </seg>
</u>
```

In this example, recoded from a corpus of language-impaired speech prepared by Fletcher and Garman, the speaker's utterance has been fully segmented into clausal (`type=C`) or minor (`type=M`) units. An additional element `<paraphasia>` has been used to define a particular characteristic of this corpus for which no element exists in the TEI scheme.

See further chapter 29 ('Modifying the TEI DTD') on p. 619 for a discussion of the way in which this kind of user-defined extension of the TEI scheme may be performed and chapter 3 ('Structure of the TEI Document Type Definition') on p. 35 for the mechanisms on which it depends.

This example also uses the core elements `<omit>` and `<del>` to mark editorial decisions concerning matter completely omitted from the transcript (because of inaudibility), and words which have been transcribed but which the transcriber considers may be deleted, respectively. See further section 6.5 ('Simple Editorial Changes') on p. 140 for a discussion of these and related elements.

It is often the case that the desired segmentation does not respect utterance boundaries; for example, syntactic units may cross utterance boundaries. For a detailed discussion of this problem, and the various methods proposed by these Guidelines for handling it, see chapter 31 ('Multiple Hierarchies') on p. 633. Methods discussed there include these:

- a concurrent DTD may be defined
- "milestone" tags may be used; the special-purpose `<shift>` tag discussed in section 11.2.6 ('Shifts') on p. 258 is an extension of this method
- where several discontinuous segments are to be grouped together to form a syntactic unit (e.g. a phrasal verb with interposed complement), the `<join>` element may be used

11.3.2 Synchronization and Overlap

A major difference between spoken and written texts is the importance of the temporal dimension to the former. As a very simple example, consider the following, first as it might be represented in a playscript:

```
Jane: Have you read Vanity Fair?
Stig: Yes
Lou: (nods vigorously)
```

Let us assume that Stig and Lou respond to Jane's question before she has finished asking it — a fairly normal situation in spontaneous speech. The simplest way of representing this *overlap* would be to use the `trans` attribute previously discussed:

```
<u who=Jane>have you read Vanity Fair</u>
<u who=Stig trans=overlap>yes</u>
<!-- ... -->
```

However, this does not allow us to indicate either the extent to which Jane's utterance is overlapped, nor does it show that there are in fact three things which are synchronous: the end of Jane's utterance, Stig's whole utterance, and Lou's kinesic. To overcome these problems, more sophisticated techniques, employing the mechanisms for pointing and alignment discussed

in detail in section 14.5 ('Synchronization') on p. 364, are needed. If the additional tag set for linking has been enabled (as described in section 11.3.1 ('Segmentation') on p. 261 above), one way to represent the simple example above would be as follows:

```
<u id=u1 who=Jane>have you read Vanity
   <anchor id=a1 synch='u2 k1'> Fair</u>
<u who=Stig id=u2>yes</u>
<kinesic id=k1 who=Lou iterated=Y desc=nod>
```

For a full discussion of this and related mechanisms, section 14.5.2 ('Placing Synchronous Events in Time') on p. 366 should be consulted. The rest of the present section, which should be read in conjunction with that more detailed discussion, presents a number of ways in which these mechanisms may be applied to the specific problem of representing temporal alignment, synchrony or overlap in transcribing spoken texts.

In the simple example above, the first utterance (that with identifier u1) contains an <anchor> element, the function of which is simply to mark a point within it. The synch attribute associated with this anchor point specifies the identifiers of the other two elements which are to be synchronized with it: specifically, the second utterance (u2) and the kinesic (k1). Note that one of these elements has content and the other is empty.

This example demonstrates only a way of indicating a point within one utterance at which it can be synchronized with another utterance and a kinesic. For more complex kinds of alignment, involving possibly multiple synchronization points, an additional element is provided, known as a <timeLine>. This consists of a series of <when> elements, each representing a point in time, and bearing attributes which indicate its exact temporal position relative to other elements in the same timeline, in addition to the sequencing implied by its position within it.

For example:

```
<timeLine unit=dsec>
   <when id=P1 absolute="12:20:01:01 BST">
   <when id=P2 interval=45 since=P1>
   <when id=P6>
   <when id=P3 interval=15 since=P6>
</timeLine>
```

This timeline represents four points in time, named P1, P2, P6, and P3 (as with all attributes named id in the TEI scheme, the names must be unique within the document but have no other significance). P1 is located absolutely, at 12:20:01:01 BST. P2 is 4.5 seconds (i.e. 45 deci-seconds) later than P2 (i.e. at 12:20:46). P6 is at some unspecified time later than P2 and previous to P3 (this is implied by its position within the timeline, as no attribute values have been specified for it). The fourth point, P3, is 1.5 seconds (15 dsec) later than P6.

One or more such timelines may be specified within a spoken text, to suit the encoder's convenience. If more than one is supplied, the origin attribute may be used on each to specify which other <timeLine> element it follows. The unit attribute indicates the units used for timings given on <when> elements contained by the alignment map. Alternatively, to avoid the need to specify times explicitly, the interval attribute may be used to indicate that all the <when> elements in a time line are a fixed distance apart.

Three methods are available for aligning points or elements within a spoken text with the points in time defined by the <timeLine>:

- The elements to be synchronized may specify the identifier of a <when> element as the value of one of the start, end or synch attributes
- The <when> element may specify the identifiers of all the elements to be synchronized with it using the synch attribute
- A free-standing <link> element may be used to associate the <when> element and the elements synchronized with it by specifying their identifiers as values for its target attribute.

For example, using the timeline given above:

```
<u id=U1 start=P2 end=P3>
    This is my <anchor synch=P6> turn
 </u>
```

The start of this utterance is aligned with P2 and its end with P3. The transition between the words 'my' and 'turn' occurs at point P6.

The synchronization represented by the preceding examples could equally well be represented as follows:

```
<timeLine units=dsec>
 <when id=P1 absolute="12:20:01:01 BST">
 <when id=P2 interval=45 since=P1 synch=U1>
 <when id=P6 synch=X1>
 <when id=P3 interval=15 since=P6 synch=U1>
</timeLine>
...
<u id=U1>This is my <anchor id=X1> turn
```

Here, the whole of the object with identifier U1 (the utterance) has been aligned with two different points, P2 and P3. This is interpreted to mean that the utterance spans at least those two points.

Finally, a <linkGrp> may be used as an alternative to the synch attribute:

```
<timeLine units=dsec>
  <when id=P1 absolute="12:20:01:01 BST">
  <when id=P2 interval=45 since=P1>
  <when id=P6>
  <when id=P3 interval=15 since=P6>
</timeLine>
...
<u id=U1><anchor id=U1start>
         This is my <anchor id=X1> turn
         <anchor id=U1end>
...
<linkGrp type=synchronous>
    <link targets='U1start P1'>
    <link targets='U1end P2'>
    <link targets='X1 P6'>
</linkGrp>
```

As a further example of the three possibilities, consider the following dialogue, represented first as it might appear in a conventional playscript:

```
Tom: I used to smoke - -
Bob: (interrupting) You used to smoke?
Tom: (at the same time) a lot more than this.  But I never
     inhaled the smoke
```

A commonly used convention might be to transcribe such a passage as follows:

```
<1> I used to smoke [ a lot more than this ]
<2>                 [ you used to smoke ]
<1> but I never inhaled the smoke
```

Such conventions have the drawback that they are hard to generalize or to extend beyond the very simple case presented here. Their reliance on the accidentals of physical layout may also make them difficult to transport and to process computationally. These Guidelines recommend one of the courses described in what follows:

Where the whole of one or another utterance is to be synchronized, the start and end attributes may be used:

```
<u who=TOM>I used to smoke <anchor id=P1> a lot more than this
     <anchor id=P2>but I never inhaled the smoke</u>
<u who=BOB start=P1 end=P2>You used to smoke</u>
```

Note that the second utterance above could equally well be encoded as follows with exactly the same effect:

```
<u who=BOB><anchor synch=P1>You used to smoke<anchor synch=P2></u>
```

If synchronization with specific timing information is required, a <timeLine> must be included:

```
<timeLine>
   <when id=T1>
   <when id=T2>
</timeLine>
  ....
<u who=TOM>I used to smoke
     <anchor synch=T1>a lot more than this
     <anchor synch=T2>but I never inhaled the smoke</u>
<u who=BOB>
     <anchor synch=T1>You used to smoke<anchor synch=T2>
</u>
```

As above, since the whole of Bob's utterance is to be aligned, the start and end attributes may be used as an alternative to the second pair of <anchor> elements:

```
<!-- ... -->
<u who=BOB start=T1 end=T2>You used to smoke</u>
```

An alternative approach is to mark the synchronization by pointing from the <timeLine> to the text:

```
<timeLine>
   <when id=T1 synch='N1 U2'>
   <when id=T2 synch='N2 U2'>
</timeLine>
   ....
<u who=TOM>I used to smoke
     <anchor ID=N1> a lot more than this
     <anchor ID=N2>but I never inhaled the smoke</u>
<u id=U2 who=BOB>You used to smoke</u>
```

To avoid deciding whether to point from the timeline to the text or vice versa, a <linkGrp> may be used:

```
<timeLine>
   <when id=T1>
   <when id=T2>
</timeLine>
<!-- ... -->
<u who=TOM>I used to smoke
     <anchor ID=N1> a lot more than this
     <anchor ID=N2>but I never inhaled the smoke</u>
<u id=U2 who=BOB>You used to smoke</u>
!--- ... -->
<linkGrp type=synchronize>
   <link targets='T1 N1 U2'>
   <link targets='T2 N2 U2'>
 </linkGrp>
```

Note that in each case, although Bob's utterance follows Tom's sequentially in the text, it is aligned temporally with its middle, without any need to disrupt the normal syntax of the text.

As a final example, consider the following exchange, first as it might be represented using a musical-score-like notation, in which points of synchronization are represented by vertical alignment of the text:

```
A : This is |my  |turn
B :         |Balderdash
C :         |No, |it's mine
```

All three speakers are simultaneous at the words 'my', 'Balderdash', and 'No'; speakers A and C are simultaneous at the words 'turn' and 'it's'. This could be encoded as follows, using pointers from the alignment map into the text:

```
<timeLine>
    <when id=P1 synch='A1 B1 C1'>
```

```
    <when id=P2 synch='A2 C2'>
</timeLine>
   ...
<u who=A>this is <anchor id=A1> my <anchor id=A2> turn</u>
<u who=B id=B1>balderdash</u>
<u who=C id=C1> no <anchor id=C2> it's mine</u></u>
```

11.3.3 Regularization of Word Forms

When speech is transcribed using ordinary orthographic notation, as is customary, some compromise must be made between the sounds produced and conventional orthography. Particularly when dealing with informal, dialectal or other varieties of language, the transcriber will frequently have to decide whether a particular sound is to be treated as a distinct vocabulary item or not. For example, while in a given project 'kinda' may not be worth distinguishing as a vocabulary item from 'kind of', 'isn't' may clearly be worth distinguishing from 'is not'; for some purposes, the regional variant 'isnae' might also be worth distinguishing in the same way.

One rule of thumb might be to allow such variation only where a generally accepted orthographic form exists, for example, in published dictionaries of the language register being encoded; this has the disadvantage that such dictionaries may not exist. Another is to maintain a controlled (but extensible) set of normalized forms for all such words; this has the advantage of enforcing some degree of consistency among different transcribers. Occasionally, as for example when transcribing abbreviations or acronyms, it may be felt necessary to depart from conventional spelling to distinguish between cases where the abbreviation is spelled out letter by letter (e.g. 'B B C' or 'V A T') and where it is pronounced as a single word ('VAT' or 'RADA'). Similar considerations might apply to pronunciation of foreign words (e.g. 'Monsewer' vs. 'Monsieur').

In general, use of punctuation, capitalization, etc., in spoken transcripts should be carefully controlled. It is important to distinguish the transcriber's intuition as to what the punctuation should be from the marking of prosodic features such as pausing, intonation, etc.

Whatever practice is adopted, it is essential that it be clearly and fully documented in the editorial declarations section of the header. It may also be found helpful to include normalized forms of non-conventional spellings within the text, using the elements for simple editorial changes described in section 6.5 ('Simple Editorial Changes') on p. 140 (see further section 11.3.5 ('Speech Management') on p. 267).

11.3.4 Prosody

In the absence of conventional punctuation, the marking of prosodic features assumes paramount importance, since these structure and organize the spoken message. Indeed, such prosodic features as points of primary or secondary stress may be represented by specialized punctuation marks. Pauses have already been dealt with in section 11.2.2 ('Pause') on p. 255; while tone units (or intonational phrases) can be indicated by the segmentation tag discussed in section 11.3.1 ('Segmentation') on p. 261. The shift tag discussed in section 11.2.6 ('Shifts') on p. 258 may also be used to encode some prosodic features, for example where all that is required is the ability to record shifts in voice quality.

For more detailed work, involving a detailed phonological transcript including representation of stress and pitch patterns, it is probably best to maintain the prosodic description in parallel with the conventional written transcript, rather than attempt to embed detailed prosodic information within it. The two parallel streams may be aligned with each other and with other streams, for example an acoustic encoding, using the general alignment mechanisms discussed in section 11.2.6 ('Shifts') on p. 258.

Where only a small number of phonetic or phonemic aspects are included in a transcript, it may be convenient to provide a simple set of entity declarations for the particular set of features marked. The entity references in the text may then be redefined to produce simple punctuation marks (as in the following example), or as references to bundles of phonological features, in the same way as is proposed for part of speech tags (see section 15.4 ('Linguistic Annotation') on p. 392).

In the following example, a small set of prosodic features are recorded throughout the transcript using a user-defined entity set such as the following:

```
<!ENTITY lf ".">    <!-- low fall intonation -->
<!ENTITY fr ",">    <!-- fall rise intonation -->
<!ENTITY lr "?">    <!-- low rise intonation -->
<!ENTITY rf "!">    <!-- rise fall intonation -->
<!ENTITY trunc "-"> <!-- truncated syllable -->
<!ENTITY long ":">  <!-- lengthened syllable -->
```

This set of entity definitions may be included directly within the document type declaration subset for the file, or more conveniently along with any other extensions or modifications within the user extensions file defined by the entity *TEI.extensions.ent*, as discussed in section 3.3 ('Invocation of the TEI DTD') on p. 39. For convenience of reading on the screen, these entity declarations will map the mnemonic entity names used in the text below to a conventional punctuation mark.

```
<div n="Lod E-03">
<note>C is with a friend</note>
<u who=Cwn>
   <unclear>Excuse me&lf;</unclear><pause>
   You dont have some aesthetic&trunc;
   <pause><unclear>specially on early</unclear>
   aesthetics terminology &lr;</u>
<u who=AJ>No&lf; <pause>No&lf;
   <omit extent='2'> I'm afraid&lf;</u>
<u who=Cwn trans=latching>No&lr;
    <unclear>Well</unclear> thanks&lr; <pause>
    Oh&trunc; <unclear>you couldnt&trunc can we</unclear>
    kind of&long; <pause>I mean ask you to order
    it for us&long;&fr;</u>
<u who=AJ trans=latching>Yes&fr;
   if you know the title&lf; Yeah&lf;</u>
<u who=Cwn>
   <omit extent='3'>
   <omit extent='4'></u>
<u who=AJ>Yes thats fine.
    <unclear>just as soon as it comes in we'll send you
    a postcard&lf</unclear></u>
```

This example, which is taken from a corpus of bookshop service encounters[6] also demonstrates the use of the <unclear> and <omit> elements discussed in section 6.5 ('Simple Editorial Changes') on p. 140. Where words are so unclear that only their extent can be recorded, the empty <omit> element may be used; where the encoder can identify the words but wishes to record a degree of uncertainty about their accuracy, the <unclear> element may be used. More flexible and detailed methods of indicating uncertainty are discussed in chapter 17 ('Certainty and Responsibility') on p. 435.

Where a transcript includes many phonetic or phonemic aspects, it will generally be convenient to use a specialized writing system, as defined in chapters 4 ('Characters and Character Sets') on p. 71 and 25 ('Writing System Declaration') on p. 571. For representation of phonemic information, the use of the International Phonetic Alphabet is recommended.

11.3.5 Speech Management

Phenomena of *speech management* include disfluencies such as filled and unfilled pauses, interrupted or repeated words, corrections, and reformulations as well as interactional devices asking for or providing feedback. Depending on the importance attached to such features, transcribers may choose to adopt conventionalized representations for them (as discussed in section 11.3.3 ('Regularization of Word Forms') on p. 266 above), or to transcribe them using IPA or some other

[6] Laura Gavioli and Gillian Mansfield, *The Pixi Corpora* (Bologna: Cooperativa Libraria Universitaria Editrice, 1990), p. 74.

transcription system. To simplify analysis of the lexical features of a speech transcript, it may be felt useful to "tidy away" many of these disfluencies. Where this policy has been adopted, these Guidelines recommend the use of the tags for simple editorial intervention discussed in section 6.5 ('Simple Editorial Changes') on p. 140, to make explicit the extent of regularization or normalization performed by the transcriber.

For example, false starts, repetition, and truncated words might all be included within a transcript, but marked as editorially deleted, in the following way:

```
<del type=truncation>s</del>see
<del type=repetition>you you</del>you know
<del type=falseStart>it's</del>he's crazy
```

As previously noted, the <gap> element may be used to mark points within a transcript where words have been omitted, for example because they are inaudible:

```
<omit extent='approx 10 sylls'
      reason='passing truck'>
```

The <unclear> element may be used to mark words which have been included although the transcriber is unsure of their accuracy:

```
and then <unclear cause='passing truck'>marbled queen</unclear>
```

Where a transcriber is believed to have incorrectly identified a word, the elements <corr> or <sic> may be used to indicate both the original and a corrected form of it:

```
<sic corr="SCSI" resp=DD>skuzzy</sic>
<corr sic="skuzzy" resp=AGB>SCSI</corr>
```

As discussed in section 6.5.1 ('Correction of Apparent Errors') on p. 141, the first of these would be appropriate where faithfulness to the transcribers' intuition is paramount, and the second where the editorial interpretation is felt more significant. In either case, the user of the text can perceive the basis of the choice being offered.

11.3.6 Analytic Coding

The recommendations made here only concern the establishment of a basic text. Where a more sophisticated analysis is needed, more sophisticated methods of markup will also be appropriate, for example, using concurrent markup streams to indicate multiple segmentation of the stream of discourse, or complex alignment of several segments within it. Where additional annotations (sometimes called "codes" or "tags") are used to represent such features as linguistic word class (noun, verb, etc.), type of speech act (imperative, concessive, etc.), or information status (theme/rheme, given/new, active/semi-active/new), etc., a selection from the general purpose analytic tools discussed in chapters 14 ('Linking, Segmentation, and Alignment') on p. 331, 15 ('Simple Analytic Mechanisms') on p. 381, and 16 ('Feature Structures') on p. 397, may be used to advantage.

Chapter 12

Print Dictionaries

This chapter defines a base tag set for encoding human-oriented monolingual and polyglot dictionaries (as opposed to computational lexica, which are intended for use by language-processing software). Dictionaries are most familiar in their printed form; however, increasing numbers of dictionaries exist also in electronic forms which are independent of any particular printed form, but from which various displays can be produced — e.g. CD-ROM dictionaries.

Both typographically and structurally, dictionaries are extremely complex. In addition, dictionaries interest many communities with different and sometimes conflicting goals. As a result, many general problems of text encoding are particularly pronounced here, and more compromises and alternatives within the encoding scheme may be required. [1]

Two problems are particularly prominent.

First, because the structure of dictionary entries varies widely both among and within dictionaries, the simplest way for an encoding scheme to accommodate the entire range of structures actually encountered is to allow virtually any element to appear virtually anywhere in a dictionary entry. It is clear, however, that strong and consistent structural principles do govern the vast majority of conventional dictionaries, as well as many or most entries even in more "exotic" dictionaries; ideally, a set of encoding guidelines should capture these structural principles. We therefore define two distinct elements for dictionary entries, one (<entry>) which captures the regularities of most conventional dictionary entries, and a second (<entryFree>) which uses the same elements, but allows them to combine much more freely. It is recommended that <entry> be used in preference to <entryFree> wherever the structure of the entry allows it. These elements and their contents are described in sections 12.2 ('The Structure of Dictionary Entries') on p. 274, 12.6 ('Unstructured Entries') on p. 308, and 12.4 ('Headword and Pronunciation References') on p. 297.

Second, since so much of the information in printed dictionaries is implicit or highly compressed, their encoding requires clear thought about whether it is to capture the precise typographic form of the source text or the underlying structure of the information it presents. Since both of these views of the dictionary may be of interest, it proves necessary to develop

[1] We refer the reader to previous and current discussions of a common format for encoding dictionaries. For example,
Robert A. Amsler and Frank W. Tompa, *An SGML-Based Standard for English Monolingual Dictionaries,* in *Information in Text: Fourth Annual Conference of the U[niversity of] W[aterloo] Centre for the New Oxford English Dictionary* October 26-28, 1988, Waterloo, Canada, pp. 61-79;
Nicoletta Calzolari et al., *Computational Model of the Dictionary Entry: Preliminary Report,* Acquilex: Esprit Basic Research Action No. 3030, Six-Month Deliverable, Pisa, April 1990;
John Fought and Carol Van Ess-Dykema, *Toward an SGML Document Type Definition for Bilingual Dictionaries,* TEI working paper TEI AIW20 (available from the TEI);
Nancy Ide and Jean Veronis, *Encoding Print Dictionaries*, *Computers and the Humanities* (special TEI issue — to appear);
Nancy Ide, Jacques Le Maitre, and Jean Veronis, *Outline of a Model for Lexical Databases,* (Information Processing and Management, 29, 2, 159-186, 1993);
Nancy Ide, Jean Veronis, Susan Warwick- Armstrong, Nicoletta Calzolari, *Principles for Encoding machine readable dictionaries*, *Proceedings of the Fifth EURALEX International Congress, EURALEX'92* (to appear), University of Tempere, Finland; and
The DANLEX Group, *Descriptive tools for electronic processing of dictionary data,* in *Lexicographica, Series Maior* (Tübingen: Niemeyer, 1987).

methods of recording both, and of recording the interrelationship between them as well. Users interested mainly in the printed format of the dictionary will require an encoding to be faithful to an original printed version. However, other users will be interested primarily in capturing the lexical information in a dictionary in a form suitable for further processing, which may demand the expansion or rearrangement of the information contained in the printed form. Further, some users wish to encode *both* of these views of the data, and retain the links between related elements of the two encodings. Problems of recording these two different views of dictionary data are discussed in section 12.5 ('Typographic and Lexical Information in Dictionary Data') on p. 300, together with mechanisms for retaining both views when this is desired.

Whichever view is adopted, a parameter entity *TEI.dictionaries* must be declared within the document type subset of any document using this base tag set. This should have the value INCLUDE, as further described in section 3.3 ('Invocation of the TEI DTD') on p. 39. A document using this base tag set and no other additional tag sets will thus begin as follows:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
     <!ENTITY % TEI.dictionaries 'INCLUDE' >
]>
```

12.1 Dictionary Body and Overall Structure

Overall, dictionaries have the same structure of front matter, body, and back matter familiar from other texts; the base tag set for dictionaries uses the same front-matter and back-matter elements as other TEI base tag sets; these are documented in chapter 7 ('Default Text Structure') on p. 183. In addition, dictionaries define the elements <entry>, <entryFree>, and <superEntry> as component-level elements which can occur directly within a text division or the text body.

The following tags should be used to mark the gross structure of a printed dictionary; the dictionary-specific tags are discussed further in the following section.

<text> contains a single text of any kind, whether unitary or composite, for example a poem or drama, a collection of essays, a novel, a dictionary, or a corpus sample.

<front> contains any prefatory matter (headers, title page, prefaces, dedications, etc.) found before the start of a text proper.

<body> contains the whole body of a single unitary text, excluding any front or back matter.

<back> contains any appendixes, etc. following the main part of a text.

** ** contains a subdivision of the front, body, or back of a text.

<div0> contains the largest possible subdivision of the body of a text.

<div1> contains a first-level subdivision of the front, body, or back of a text (the largest, if <div0> is not used, the second largest if it is).

<entry> contains a reasonably well-structured dictionary entry.

<entryFree> contains a dictionary entry which does not necessarily conform to the constraints imposed by the <entry> element.

<superentry> groups successive entries for a set of homographs.

The text-division elements <div2> through <div7> may also be used, as described in chapter 7 ('Default Text Structure') on p. 183.

As members of the class *entries*, <entry> and <entryFree> share the following attributes:

type indicates type of entry, in dictionaries with multiple types. Suggested values include:

- ***main*** a main entry (default).
- ***hom*** a homograph with a separate entry.
- ***xref*** a reduced entry whose only function is to point to another main entry (e.g. for forms of an irregular verb or for variant spellings: 'was' pointing to 'be', or 'esthete' to 'aesthete').
- ***affix*** an entry for a prefix, infix, or suffix.
- ***abbr*** an entry for an abbreviation.

supplemental a supplemental entry (for use in dictionaries which issue supplements to their main work in which they include updated information about entries).

foreign an entry for a "foreign" word in a monolingual dictionary.

key contains a (sortable) character sequence reflecting the entry's alphabetical position in the printed dictionary.

The front and back matter of a dictionary may well contain specialized material like lists of common and proper nouns, grammatical tables, gazetteers, a "guide to the use of the dictionary", etc. These may be tagged as elements defined in the core tag set (chapter 6 ('Elements Available in All TEI Documents') on p. 119) or as specialized dictionary elements as defined in this chapter.

The <body> element consists of a set of *entries*, optionally grouped into one or several , <div0>, or <div1> elements. These text divisions might correspond, for example, to sections for different languages in a bilingual dictionaries, sections for different letters of the alphabet, etc.[2] In print dictionaries, entries are typically typographically distinct entities, each headed by some morphological form of the lexical item described (the *headword*), and sorted in alphabetical order or (for non-alphabetic scripts) in some other conventional sequence. Dictionary entries should be encoded as distinct successive items, each marked as an <entry> element. The type attribute may be used to distinguish different types of entries, for example main entries, related entries, run-on entries, or entries for cross-references, etc.

Some dictionaries provide distinct entries for homographs, on the basis of etymology, part-of-speech, or both, and typically provide a numeric superscript on the headword identifying the homograph number. In these cases each homograph should be encoded as a separate entry; the <superEntry> element may optionally be used to group such successive homograph entries. In addition to a series of <entry> elements, the <superEntry> may contain a preliminary <form> group (see section 12.3.1 ('Information on Written and Spoken Forms') on p. 279) when information about hyphenation, pronunciation, etc., is given only once for two or more homograph entries. If the homograph number is to be recorded, the global attribute n should be used for this purpose. In some dictionaries, homographs are treated in distinct parts of the same entry; in these cases, they may be separated by use of the <hom> element, for which see section 12.2.1 ('Hierarchical Levels') on p. 274.

A sort key, given in the key attribute, is often required for superentries and entries, especially in cases where the order of entries does not follow the local character-set collating sequence (as, for example, when an entry for "3D" appears at the place where "three-D" would appear).

The body of a bilingual dictionary with two parts will thus have an overall structure resembling the following:

```
<body>
  <div0 type='dictionary'>
    <!-- English-French -->
    <entry>...</entry>
    <entry>...</entry>
    <entry>...</entry>
  <!-- ... -->
  </div0>
  <div0>
    <!-- French-English -->
    <entry>...</entry>
    <entry>...</entry>
    <entry>...</entry>
  <!-- ... -->
  </div0>
</body>
```

A dictionary with no internal divisions might have a structure like the following; a <superEntry> is shown grouping two homograph entries.

```
<body>
  <entry>...</entry>
```

[2] It is unlikely that many conventional dictionaries will require smaller divisions, but all the usual division elements <div2> through <div7> may be used.

```
  <entry>...</entry>
  <!-- ... -->
  <superEntry>
    <entry type=hom n='1'>...</entry>
    <entry type=hom n='2'>...</entry>
  </superEntry>
  <!-- ... -->
</body>
```

The base tag set for dictionaries is contained in the files *teidict2.ent* and *teidict2.dtd*. The first of these defines the class *comp.dictionaries*, so that the generic text-division elements , <div0>, <div1>, etc. can contain <entry> elements:

```
<!-- 12.1:  Element classes for dictionary base              -->
<!-- Text Encoding Initiative: Guidelines for Electronic     -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.   -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy   -->
<!-- in any form is granted, provided this notice is         -->
<!-- included in all copies.                                 -->

<!-- These materials may not be altered; modifications to    -->
<!-- these DTDs should be performed as specified in the      -->
<!-- Guidelines in chapter "Modifying the TEI DTD."          -->

<!-- These materials subject to revision. Current versions   -->
<!-- are available from the Text Encoding Initiative.        -->


<!-- First we define attributes available on all the         -->
<!-- elements in this tag set.                               -->

<!-- ... declarations from section 12.5.4                    -->
<!--     (Attributes for dictionary work)                    -->
<!--     go here ...                                         -->

<!-- Next we define comp.dictionaries, which will be used in -->
<!-- the declaration of component, within file TEI2.DTD.     -->

<!ENTITY % x.comp.dictionaries ''                             >
<!ENTITY % m.comp.dictionaries '%x.comp.dictionaries entry |
          entryFree | superentry'                             >
<!ENTITY % mix.dictionaries '| %m.comp.dictionaries'          >

<!-- Next, we declare some specialized element classes, used -->
<!-- in various content models in the dictionary tag set.    -->

<!ENTITY % a.entries '
          type              CDATA              "main"
          key               CDATA              #IMPLIED'     >

<!-- ... declarations from section 12.2.2                    -->
<!--  (Class for top-level structure of dictionary entries)  -->
<!--     go here ...                                         -->
<!-- ... declarations from section 12.3.1                    -->
<!--     (Classes for morphological and form information)    -->
<!--     go here ...                                         -->
<!-- ... declarations from section 12.3.2                    -->
<!--     (Elements for grammatical information)              -->
<!--     go here ...                                         -->
<!-- ... declarations from section 12.4                      -->
<!--     (Classes for headword references)                   -->
```

```
<!--     go here ...                                          -->
<!-- ... declarations from section 12.6                       -->
<!--     (Model class for unstructured dictionary entries)    -->
<!--     go here ...                                          -->
```

The dictionary-specific elements are all declared in the file *teidict2.dtd*, which has the following overall structure.

```
<!-- 12.1:  Base tag set for printed dictionaries             -->
<!-- Text Encoding Initiative: Guidelines for Electronic      -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.    -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy    -->
<!-- in any form is granted, provided this notice is          -->
<!-- included in all copies.                                  -->

<!-- These materials may not be altered; modifications to     -->
<!-- these DTDs should be performed as specified in the       -->
<!-- Guidelines in chapter "Modifying the TEI DTD."           -->

<!-- These materials subject to revision. Current versions    -->
<!-- are available from the Text Encoding Initiative.         -->

<!-- First we embed the default text structure.               -->

<![ %TEI.singleBase [
<!ENTITY % TEI.structure.dtd system 'teistr2.dtd'              >
%TEI.structure.dtd;
]]&nil;>


<!-- Now we define the dictionary-specific material.          -->

<!-- ... declarations from section 12.2.1                     -->
<!--     (Dictionary entries and their structure)             -->
<!--     go here ...                                          -->
<!-- ... declarations from section 12.3.1                     -->
<!--     (The form group)                                     -->
<!--     go here ...                                          -->
<!-- ... declarations from section 12.3.2                     -->
<!--     (The gram group)                                     -->
<!--     go here ...                                          -->
<!-- ... declarations from section 12.3.3.1                   -->
<!--     (Definition text)                                    -->
<!--     go here ...                                          -->
<!-- ... declarations from section 12.3.3.2                   -->
<!--     (Translation information)                            -->
<!--     go here ...                                          -->
<!-- ... declarations from section 12.3.4                     -->
<!--     (Etymologies)                                        -->
<!--     go here ...                                          -->
<!-- ... declarations from section 12.3.5.1                   -->
<!--     (Examples and citations)                             -->
<!--     go here ...                                          -->
<!-- ... declarations from section 12.3.5.2                   -->
<!--     (Usage information)                                  -->
<!--     go here ...                                          -->
<!-- ... declarations from section 12.3.5.3                   -->
<!--     (Cross References)                                   -->
<!--     go here ...                                          -->
<!-- ... declarations from section 12.3.6                     -->
<!--     (Related entries)                                    -->
```

```
<!--     go here ...                                  -->
<!-- ... declarations from section 12.4               -->
<!--     (Headword references)                        -->
<!--     go here ...                                  -->
```

12.2 The Structure of Dictionary Entries

A simple dictionary entry may contain information about the form of the word treated, its grammatical characterization, its definition, synonyms, or translation equivalents, its etymology, cross-references to other entries, usage information, and examples. These we refer to as the *constituent parts* or *constituents* of the entry; some dictionary constituents possess no internal structure, while others are most naturally viewed as groups of smaller elements, which may be marked in their own right. In some styles of markup, tags will be applied only to the low-level items, leaving the constituent groups which contain them untagged. We distinguish the class of *top-level constituents* of dictionary entries, which can occur directly within entries, from the class of *phrase-level* constituents, which can normally occur only within top-level constituents. The top-level constituents of dictionary entries are described in section 12.2.2 ('Groups and Constituents') on p. 276, and documented more fully, together with their phrase-level sub-constituents, in section 12.3 ('Top-level Constituents of Entries') on p. 279.

In addition, however, dictionary entries often have a complex hierarchical structure. For example, an entry may consist of two or more sub-parts, each corresponding to information for a different part-of-speech homograph of the headword. The entry (or part-of-speech homographs, if the entry is split this way) may also consist of senses, each of which may in turn be composed of two or more sub-senses, etc. Each sub-part, homograph entry, sense, or sub-sense we call a *level*; at any level in an entry, any or all of the constituent parts of dictionary entries may appear. The hierarchical levels of dictionary entries are documented in section 12.2.1 ('Hierarchical Levels') on p. 274.

12.2.1 Hierarchical Levels

The outermost structural level of an entry is marked with the elements <entry> or <entryFree>. The <hom> element marks the subdivision of entries into part-of-speech homographs. The <sense> element marks the subdivision of entries and part-of-speech homographs into senses; this element nests recursively in order to provide for a hierarchy of sub-senses of any depth. All of these levels may each contain any of the constituent parts of an entry. A special case of hierarchical structure is represented by the <re> (related entry) element, which is discussed in section 12.3.6 ('Related Entries') on p. 296.

<entry> contains a reasonably well-structured dictionary entry.

<entryFree> contains a dictionary entry which does not necessarily conform to the constraints imposed by the <entry> element.

<hom> groups information relating to one homograph within an <entry>.

<sense> groups together all information relating to one word sense in a dictionary <entry> (definitions, examples, translation equivalents, etc.) Attributes include:

level gives the nesting depth of this sense.

For example, an entry with two senses will have the following structure:

```
<entry>
  <!-- ... information common to both senses -->
  <sense n='1'>
     <!-- ... sense number 1 -->
  </sense>
  <sense n='2'>
     <!-- ... sense number 2 -->
  </sense>
</entry>
```

An entry with two homographs, the first with two senses and the second with three (one of which has two sub-senses), will have a structure like this:

```
<entry>
  <!-- ... information common to both homographs, if any ... -->
  <hom n='1'>
    <sense n='1'>...</sense>
    <sense n='2'>...</sense>
  </hom>
  <hom n='2'>
    <sense n='1'>
      <sense n='a'>...</sense>
      <sense n='b'>...</sense>
    </sense>
    <sense n='2'>...</sense>
    <sense n='3'>...</sense>
  </hom>
</entry>
```

In some dictionaries, homographs typically receive separate entries; in such a case, as noted in section 12.1 ('Dictionary Body and Overall Structure') on p. 270, the two homographs may be treated as entries, optionally grouped by a superentry:

```
<superEntry>
<!-- ... form information common to both
     homographs, if any ... -->
<entry n='1'>
  <sense n='1'>...</sense>
  <sense n='2'>...</sense>
</entry>
<entry n='2'>
  <sense n='1'>
    <sense n='a'>...</sense>
    <sense n='b'>...</sense>
  </sense>
  <sense n='2'>...</sense>
  <sense n='3'>...</sense>
</entry>
</superEntry>
```

The hierarchical levels of dictionary entries are declared as shown in the following DTD fragment. As may be seen, the content model for <entry> specifies that entries do not nest, that homographs nest within entries, and that senses nest within entries, homographs, or senses, and may be nested to any depth to reflect the embedding of sub-senses. Any of the top-level constituents (<def>, <usg>, <form>, etc.) can appear at any level (i.e., within entries, homographs, or senses).

```
<!-- 12.2.1:  Dictionary entries and their structure          -->
<!ELEMENT superentry    - O  (form?, entry+)                    >
<!ATTLIST superentry         %a.global;
                             %a.entries;                        >
<!ELEMENT entry         - O  (hom | sense |
                             %m.dictionaryTopLevel)+
                                          +(anchor)     >
<!ATTLIST entry              %a.global;
                             %a.entries;                        >
<!ELEMENT entryFree     - O  (#PCDATA)

                                         +(%m.dictionaryParts
                                         | %m.phrase |
                                         %m.inter)      >
<!ATTLIST entryFree          %a.global;
                             %a.dictionaries;
```

```
                              %a.entries;                          >
<!ELEMENT hom           - O   (sense | %m.dictionaryTopLevel)*
                                                  -(entry)    >
<!ATTLIST hom                 %a.global;
                              %a.dictionaries;                     >
<!ELEMENT sense         - -   (sense | %m.dictionaryTopLevel |
                              %m.phrase | #PCDATA)*                >
<!ATTLIST sense               %a.global;
                              %a.dictionaries;
          level               NUMBER              #IMPLIED       >
<!-- This fragment is used in sec. 12.1                          -->
```

12.2.2 Groups and Constituents

As noted above, dictionary entries, and subordinate levels within dictionary entries, may comprise several constituent parts, each providing a different type of information about the word treated. The *top-level constituents* of dictionary entries are:

- information about the form of the word treated (orthography, pronunciation, hyphenation, etc.)
- grammatical information (part of speech, grammatical sub-categorization, etc.)
- definitions or translations into another language
- etymology
- examples
- usage information
- cross-references to other entries
- notes
- entries (often of reduced form) for related words, typically called *related entries*

Any of the hierarchical levels (<entry>, <entryFree>, <hom>, <sense>) may contain any of these top-level constituents, since information about word form, particular grammatical information, special pronunciation, usage information, etc., may apply to an entire entry, or to only one homograph, or only to a particular sense. The examples below illustrate this point.

The following elements are used to encode these top-level constituents:

<form> groups all the information on the written and spoken forms of one headword.

<gramGrp> groups morpho-syntactic information about a lexical item, e.g. <pos>, <gen>, <number>, <case>, or <itype> (inflectional class).

<def> contains definition text in a dictionary entry.

<trans> contains translation text and related information (within an entry in a multilingual dictionary).

<eg> (in a dictionary) contains an example text containing at least one occurrence of the word form, used in the sense being described; examples may be quoted from (named) authors or contrived.

<usg> contains usage information in a dictionary entry.

<xr> contains a phrase, sentence, or icon referring the reader to some other location in this or another text.

<etym> encloses the etymological information in a dictionary entry.

<re> contains a dictionary entry for a lexical item related to the headword, such as a compound phrase or derived form, embedded inside a larger entry.

<note> contains a note or annotation.

In a simple entry with no internal hierarchy, all top-level constituents appear at the <entry> level.[3]

[3] Each example taken from a real dictionary indicates its source using the following abbreviations for dictionary names:

C/R Beryl T. Atkins et al., *Collins Robert French-English English-French Dictionary* (London: Collins, 1978, rpt. 1983)

CED Collins English Dictionary

CP Collins Pocket

DNT *Le Dictionnaire de Notre Temps 1990*, ed. Franc,,oise Guerard (Paris: Hachette, 1990).

" **com.peti.tor** /k@m"petit@(r)/ n person who competes. [OALD] "

```
<entry>
  <form>
    <orth>competitor</orth>
    <hyph>com|peti|tor</hyph>
    <pron>k@m"petit@(r)</pron>
  </form>
  <gramGrp>
    <pos>n</pos>
  </gramGrp>
  <def>person who competes.</def>
</entry>
```

For the elements which appear within the <form> and <gramGrp> elements of this example, see below, section 12.3.1 ('Information on Written and Spoken Forms') on p. 279, and section 12.3.2 ('Grammatical Information') on p. 284.

As mentioned above, any top-level constituent can appear at any level when the hierarchical structure of the entry is more complex. The most obvious examples are <def> and <trans>, which appear at the <sense> level when several senses or translations exist:

" **disproof** (dIs"pru:f) n. 1. facts that disprove something. 2. the act of disproving. [CED] "

```
<entry>
  <form>
    <orth>disproof</orth>
    <pron>dIs"pru:f</pron>
  </form>
  <gramGrp><pos>n</></gramGrp>
  <sense n='1'><def>facts that disprove something.</def></sense>
  <sense n='2'><def>the act of disproving.</def></sense>
</entry>
```

In the following example, <gramGrp> is used to distinguish two homographs:

" **bray** /breI/ n cry of an ass; sound of a trumpet. ■ vt [VP2A] make a cry or sound of this kind. [OALD] "

```
<entry>
  <form>
    <orth>bray</orth>
    <pron>breI</pron>
  </form>
  <hom>
    <gramGrp><pos>n</pos></gramGrp>
    <def>cry of an ass; sound of a trumpet.</def>
```

LDOCE Longman Dictionary of Contemporary English

NPEG *The New Penguin English Dictionary* (London: Penguin, 1986, rpt. 1987).

OALD *Oxford Advanced Learner's Dictionary of Current English*, ed. A. S. Hornby with A. P. Cowie and A. C. Gimson ([Oxford]: Oxford University Press, 1974).

PLC *Petit Larousse en Couleurs* (Paris: Larousse, 1990).

PLI *Pequeño Larousse Illustrado* por Ramo"n Garci"a-Pelayo y Gross (Buenos Aires, Me"xico, Paris: Ediciones Larousse, 1964).

PR Le Petit Robert

SSSE *Simon and Schuster's International Dictionary English/Spanish Spanish/English* ed. Tana de Gámez (New York: Simon and Schuster, 1973).

W7 Webster's 7th Collegiate

WNC *Webster's New Collegiate Dictionary* (Springfield, Mass.: G. & C. Merriam Co., 1975).

To simplify the electronic presentation of this document on systems with limited character sets, many of the pronunciations are presented using the transliteration found in the electronic edition of the *Oxford Advanced Learner's Dictionary*. Also, the middle dot in quoted entries is rendered with a full stop, while within the sample transcriptions hyphenation and syllabification points are indicated with |, regardless of their rendition in the source text.

```
  </hom>
  <hom>
    <gramGrp>
      <pos>vt</pos>
      <subc>VP2A</subc>
    </gramGrp>
    <def>make a cry or sound of this kind.</def>
  </hom>
</entry>
```

Information of the same kind can appear at different levels within the same entry; here, grammatical information occurs both at entry and homograph level.

" **ca.reen** /k@"ri:n/ vt,vi 1 [VP6A] turn (a ship) on one side for cleaning, repairing, etc. 2 [VP6A, 2A] (cause to) tilt, lean over to one side. [OALD] "

```
<entry>
  <form>
    <orth>careen</orth>
    <hyph>ca|reen</hyph>
    <pron>k@"ri:n</pron>
  </form>
  <gramGrp>
    <pos>vt</pos>
    <pos>vi</pos>
  </gramGrp>
  <sense n='1'>
    <gramGrp><subc>VP6A</subc></gramGrp>
    <def>turn (a ship) on one side for cleaning,
         repairing, etc.</def>
  </sense>
  <sense n='2'>
    <gramGrp>
      <subc>VP6A</subc>
      <subc>VP2A</subc>
    </gramGrp>
    <def>(cause to) tilt, lean over to one side.</def>
  </sense>
</entry>
```

Alone among the constituent groups, <form> can appear at the <superEntry> level as well as at the <entry>, <hom>, and <sense> levels:

" **a.ban.don** 1 /@"b&nd@n/ v [T1] 1 to leave completely and for ever; desert: The sailors abandoned the burning ship. 2 ...

abandon 2 n [U] the state when one's feelings and actions are uncontrolled; freedom from control: The people were so excited that they jumped and shouted with abandon / in gay abandon. [LDOCE] "

```
<superEntry>
  <form>
    <orth>abandon</orth>
    <hyph>a|ban|don</hyph>
    <pron>@"b&amp;nd@n</pron>
  </form>
  <entry n='1'>
    <gramGrp> <pos>v</pos> <subc>T1</subc> </gramGrp>
    <sense n='1'><def>to leave completely and for ever ...</def>
      <!-- ... -->
    </sense>
    <sense n='2'>
      <!-- ... -->
    </sense>
  </entry>
```

```
    <entry n='2'>
      <gramGrp> <pos>n/pos> <subc>U</subc> </gramGrp>
      <def>the state when one's feelings and actions are
              uncontrolled; freedom from control</def>
      <!-- ... -->
    </entry>
  </superEntry>
```

The class of top-level constituents for dictionary entries is defined by the following DTD fragment:

```
<!-- 12.2.2: Class for top-level structure of dictionary      -->
<!-- entries                                                  -->
<!ENTITY % x.dictionaryTopLevel ''                             >
<!ENTITY % m.dictionaryTopLevel '%x.dictionaryTopLevel def | eg
           | etym | form | gramGrp | note | re | trans | usg |
           xr'                                                 >
<!-- This fragment is used in sec. 12.1                       -->
```

The individual constituents are declared below, each in the section which documents it in more detail.

12.3 Top-level Constituents of Entries

This section describes the top-level constituents of dictionary entries, together with the phrase-level constituents peculiar to each.

- the <form> element, which groups orthographic information and pronunciations, is described in section 12.3.1 ('Information on Written and Spoken Forms') on p. 279
- the <gramGrp> element, which groups elements for the grammatical characterization of the headword, is described in section 12.3.2 ('Grammatical Information') on p. 284
- the <def> and <trans> elements, which describe the meaning of the headword, are described in section 12.3.3 ('Sense Information') on p. 286
- the <etym> element and its special phrase-level elements are documented in section 12.3.4 ('Etymological Information') on p. 289
- the <eg>, <usg>, <lbl>, <xr>, and <note> elements are described in section 12.3.5 ('Other Information') on p. 290
- the <re> element, which marks nested entries for related words, is described in section 12.3.1 ('Information on Written and Spoken Forms') on p. 279

12.3.1 Information on Written and Spoken Forms

Dictionary entries most often begin with information about the form of the word to which the entry applies. Typically, the orthographic form of the word, sometimes marked for syllabification, is the first item in an entry. Other information about the word, including variant or alternate forms, inflected forms, pronunciation, etc., is also often given.

The following elements should be used to encode this information: the <form> element groups one or more occurrences of any of the others; it can also be recursively nested to reflect more complex sub-grouping of information about word form(s), as shown in the examples.

<form> groups all the information on the written and spoken forms of one headword. Attributes include:

type classifies form as simple, compound, etc. Suggested values include:

simple single free lexical item

lemma the headword itself

variant a variant form

compound word formed from simple lexical items

derivative word derived from headword

inflected word in other than usual dictionary form
phrase multiple-word lexical item

<orth> gives the orthographic form of a dictionary headword. Attributes include:
type gives the type of spelling.
extent gives the extent of the orthographic information provided. Sample values include:
full full form
pref prefix
suff suffix
part partial

<pron> contains the pronunciation(s) of the word. Attributes include:
extent indicates whether the pronunciation is for whole word or part. Sample values include:
full full form
pref prefix
suff suffix
part partial

<hyph> contains a hyphenated form of a dictionary headword, or hyphenation information in some other form.

<syll> contains the syllabification of the headword.

<stress> contains the stress pattern for a dictionary headword, if given separately.

<lbl> in dictionaries, contains a label for a form, example, translation, or other piece of information, e.g. "abbreviation for", "contraction of", "literally", "approximately", "synonyms:", etc.

In addition to those listed above, the following elements, which encode morphological details of the form, may also occur within **<form>** elements:

<gram> within an entry in a dictionary or a terminological data file, contains grammatical information relating to a term, word, or form. Attributes include:
type classifies the grammatical information given according to some convenient typology — in the case of terminological information, preferably the dictionary of data element types specified in ISO WD 12 620. Suggested values include:
pos part of speech (any of the word classes to which a word may be assigned in a given language, based on form, meaning, or a combination of features, e.g. noun, verb, adjective, etc.)
gen gender (formal classification by which nouns and pronouns, and often accompanying modifiers, are grouped and inflected, or changed in form, so as to control certain syntactic relationships)
num number (e.g. singular, plural, dual, ...)
animate animate or inanimate
proper proper noun or common noun

<gen> identifies the morphological gender of a lexical item, as given in the dictionary.

<number> indicates grammatical number associated with a form, as given in a dictionary.

<case> contains grammatical case information given by a dictionary for a given form.

<per> contains an indication of the grammatical person (1st, 2d, 3d, etc.) associated with a given inflected form in a dictionary.

<tns> indicates the grammatical tense associated with a given inflected form in a dictionary.

<mood> contains information about the grammatical mood of verbs (e.g. "indicative", "subjunctive", "imperative")

<itype> indicates the inflectional class associated with a lexical item. Attributes include:
type indicates the type of indicator used to specify the inflection class, when it is necessary to distinguish between the usual abbreviated indications (e.g. "inv") and other kinds of indicators, such as special codes referring to conjugation patterns, etc. Sample values include:
abbrev abbreviated indicator
verb table coded reference to a table of verbs

Of these, the <gram> element is most general, and all of the others are synonymous with <gram> elements with appropriate values (gen, number, case, etc.) for the type attribute.

Different dictionaries use different means to mark hyphenation, syllabification, and stress, and they often use special symbols (e.g., the "middle dot" for hyphenation). Many of these symbols are defined as entities within standard ISO entity sets (see chapter 37 ('Obtaining TEI WSDs') on p. 985; others may require the definition of new entities. For pronunciation transcriptions, the user is referred to the discussion of TEI definitions for the symbols of the International Phonetic Alphabet, in chapter 37 ('Obtaining TEI WSDs') on p. 985. When alternative or additional symbols are used in the encoding, the conventions used should be fully documented in the header. (This may occur, for instance, in cases where pronunciations are not given at all in the original, but are added in the encoded text, when a simplified pronunciation transcription is used, when pronunciations in several dictionaries are normalized to a single scheme, or when pronunciation symbols are transliterated into a standard character set like ISO 646, as in the scheme used by the OALD and by some examples here.)

Among the ISO entity sets most likely to be useful in the transcription of form information are:

- ISOlat1 (Latin characters and diacritics for Western European languages)
- ISOlat2 (Eastern European languages)
- ISOpub (miscellaneous publishing and special symbols)
- ISOnum (numeric and related symbols)

In the simplest case, nothing is given but the orthography:

```
<form>
  <orth>doom-laden</orth>
</form>
<!-- [CED] -->
```

Often, however, pronunciation is given.
"**soucoupe** [sukup] ... [DNT]"

```
<form>
  <orth>soucoupe</orth>
  <pron>sukup</pron>
</form>
```

For a variety of reasons including ease of processing, it may be desired to split into separate elements information which is collapsed into a single element in the source text; orthography and hyphenation may for example be transcribed as separate elements, although given together in the source text. For a discussion of the issues involved, and of methods for retaining both the presentation form and the interpreted form, see section 12.5 ('Typographic and Lexical Information in Dictionary Data') on p. 300.

This example splits orthography and hyphenation, and adds syllabification because it differs from hyphenation:
"**ar.ea** ... [W7]"

```
<form>
   <orth>area</orth>
   <hyph>ar|ea</hyph>
   <syll>ar|e|a</syll>
</form>
```

Multiple orthographic forms may be given, e.g. to illustrate a word's inflectional pattern:
"**brag** ... vb. brags, bragging, bragged ... [CED]"

```
<form>
  <orth>brag</orth>
  <pron>br&amp;g</pron>
</form>
<gramGrp>
  <pos>vb</pos>
</gramGrp>
<form type=infl>
```

```
    <orth>brags</orth>
    <orth>bragging</orth>
    <orth>bragged</orth>
  </form>
  <!-- ... -->
```

Or the inflectional pattern may be indicated by reference to a table of paradigms, as here:
" **horrifier** [ORifje] (7) vt ... [C/R] "

```
<form>
  <orth>horrifier</orth>
  <pron>ORifje</pron>
  <itype type=vbtable>7</itype>
</form>
```

As noted, <itype> etc. are synonymous with appropriately typed instances of the general <gram> element; the last example might equally be tagged thus:

```
<form>
  <orth>horrifier</orth>
  <pron>ORifje</pron>
  <gram type='itype / vbtable'>7</gram>
</form>
```

Explanatory labels may be attached to alternate forms:
" **MTBF** *abbrev. for* mean time between failures. [CED]"

```
<entry>
  <form type=abbrev>
    <orth>MTBF</orth>
  </form>
  <form type=full>
    <lbl>abbrev. for</lbl>
    <orth>mean time between failures</orth>
  </form>
</entry>
```

When multiple orthographic forms are given, a pronunciation may be associated with all of them, as here:
"**biryani** or **biriani** (%bIrI"A:nI) ... [CED]"

```
<!-- The pronunciation is associated with both forms. -->
<form>
  <orth>biryani</orth>
  <orth>biriani</orth>
  <pron>%bIrI"A:nI</pron>
</form>
```

In other cases, different pronunciations are provided for different orthographic forms; here, the <form> element is repeated to associate the first orthographic form explicitly with the first pronunciation, and the second orthographic form with the second pronunciation:
" **mackle** ("m&k@l) or macule ("m&kju:l) ... [CED] "

```
<!-- &supschwa is a small superscript schwa -->
<form>
  <orth>mackle</orth>
  <pron>"m&amp;k&supschwa;l</pron>
</form>
<form>
  <orth>macule</orth>
  <pron>"m&amp;kju:l</pron>
</form>
```

Recursive nesting of the <form> element can preserve relations among elements that are implicit in the text. For example, in the CED entry for "hospitaller", it is clear that "U.S." is associated only with "hospitaler", but that the pronunciation applies to both forms. The following encoding preserves these relations:

" **hospitaller** or U.S. hospitaler ("hQspIt@l@) ... [CED] "

```
<form>
  <orth>hospitaller</orth>
  <form>
    <usg type=geo>U.S.</usg>
    <orth>hospitaler</orth>
  </form>
  <pron>"hQspIt@l@</pron>
</form>
```

The formal declarations for the elements of the <form> group are these:

```
<!-- 12.3.1:  The form group                                    -->
<!ELEMENT form          - -  (%m.formInfo | %paraContent)+      >
<!ATTLIST form               %a.global;
                             %a.dictionaries;
          type               CDATA               #IMPLIED      >
<!ELEMENT orth          - O  (%paraContent)                     >
<!ATTLIST orth               %a.global;
                             %a.dictionaries;
          type               CDATA               #IMPLIED
          extent             CDATA               full          >
<!ELEMENT pron          - O  (%paraContent)                     >
<!ATTLIST pron               %a.global;
                             %a.dictionaries;
          extent             CDATA               full
          notation           CDATA               #IMPLIED      >
<!ELEMENT hyph          - O  (%paraContent;)                    >
<!ATTLIST hyph               %a.global;
                             %a.dictionaries;                   >
<!ELEMENT syll          - O  (%paraContent;)                    >
<!ATTLIST syll               %a.global;
                             %a.dictionaries;                   >
<!ELEMENT stress        - O  (%paraContent;)                    >
<!ATTLIST stress             %a.global;                         >
<!-- (LBL is declared with USG, elsewhere.)                    -->


<!-- Elements for morphological information:                   -->

<!ELEMENT gram          - O  (%paraContent;)                     >
<!ATTLIST gram               %a.global;
                             %a.dictionaries;
          type               CDATA               #IMPLIED      >
<!ELEMENT gen           - -  (%paraContent)                     >
<!ATTLIST gen                %a.global;
                             %a.dictionaries;                   >
<!ELEMENT number        - -  (%paraContent;)                    >
<!ATTLIST number             %a.global;
                             %a.dictionaries;                   >
<!ELEMENT case          - -  (%paraContent;)                    >
<!ATTLIST case               %a.global;
                             %a.dictionaries;                   >
<!ELEMENT per           - O  (%paraContent;)                    >
<!ATTLIST per                %a.global;
                             %a.dictionaries;                   >
<!ELEMENT tns           - O  (%paraContent;)                    >
<!ATTLIST tns                %a.global;
                             %a.dictionaries;                   >
<!ELEMENT mood          - O  (%paraContent;)                    >
<!ATTLIST mood               %a.global;
                             %a.dictionaries;                   >
```

```
<!ELEMENT itype          - -  (%paraContent)                    >
<!ATTLIST itype               %a.global;
                              %a.dictionaries;
          type                CDATA                #IMPLIED      >
<!-- This fragment is used in sec. 12.1                         -->
```

The classes of morphological elements, and of elements allowed within the <form> group, are declared thus:

```
<!-- 12.3.1:  Classes for morphological and form information  -->
<!ENTITY % x.morphInfo ''                                       >
<!ENTITY % m.morphInfo '%x.morphInfo case | gen | gram | itype
           | mood | number | per | tns'                         >
<!ENTITY % x.formInfo ''                                        >
<!ENTITY % m.formInfo '%x.formInfo form | hyph | lbl | orth |
           pron | syll | usg'                                  >
<!-- This fragment is used in sec. 12.1                        -->
```

12.3.2 Grammatical Information

The <gramGrp> element groups grammatical information, such as part of speech, subcategorization information (e.g., syntactic patterns for verbs, count/mass distinctions for nouns), etc. It can contain any of the following elements:

<pos> indicates the part of speech assigned to a dictionary headword (noun, verb, adjective, etc.)

<subc> contains subcategorization information (transitive/intransitive, countable/non-count, etc.)

<colloc> contains a collocate of the headword.

In addition, <gramGrp> can contain any of the morphological elements defined in section 12.3.1 ('Information on Written and Spoken Forms') on p. 279 for <form>:

<gram> within an entry in a dictionary or a terminological data file, contains grammatical information relating to a term, word, or form.

<itype> indicates the inflectional class associated with a lexical item.

<gen> identifies the morphological gender of a lexical item, as given in the dictionary.

<number> indicates grammatical number associated with a form, as given in a dictionary.

<case> contains grammatical case information given by a dictionary for a given form.

<per> contains an indication of the grammatical person (1st, 2d, 3d, etc.) associated with a given inflected form in a dictionary.

<tns> indicates the grammatical tense associated with a given inflected form in a dictionary.

<mood> contains information about the grammatical mood of verbs (e.g. "indicative", "subjunctive", "imperative")

Elements conveying morphological information bear different interpretations within <gramGrp> and <form> groups, the difference being that in the <form> group, the morphological information specified pertains to the specific alternate form in question, while within <gramGrp> it applies to the headword form. For example, in the entry "**pinna** ('pIn@) n., pl. -nae (-ni:) or -nas" [CED], the word defined can be either singular or plural; the "pl." specification applies only to the inflected forms provided. Compare this with "pants (paents) pl. n.", where "pl." applies to the headword itself.

As noted above in section 12.3.1 ('Information on Written and Spoken Forms') on p. 279, the elements for morphological information are simply shorthand for the general purpose <gram> element. Consider this entry for the French word 'médire': "**médire** v.t. ind. (de) ... [PLC]"

This entry can be tagged using specialized grammatical elements:

```
<form><orth>m&eacute;dire</orth></form>
<gramGrp>
   <pos>v</pos>
   <subc>t ind</subc>
   <colloc type=prep>de</colloc>
</gramGrp>
```

Or using the <gram> element:

```
<form><orth>m&eacute;dire</orth></form>
<gramGrp>
   <gram type=pos>v</>
   <gram type=subc>t ind</>
   <gram type='colloc / prep'>de</>
</gramGrp>
```

Like <form>, <gramGrp> can be repeated, recursively nested, or used at the <sense> level to show relations among elements.

"**isotope** adj. et n. m. ... [DNT]"

```
<form><orth>isotope</orth></form>
<gramGrp>
   <pos>adj</pos>
</gramGrp>
<gramGrp>
   <pos>n</pos>
   <gen>m</gen>
</gramGrp>
```

" **wits** (wIts) pl. n. 1. (sometimes sing.) the ability to reason and act, esp. quickly... [CED] "

```
<entry>
  <form>
    <orth>wits</orth>
    <pron>wIts</pron>
  </form>
  <gramGrp>
    <number>pl</number>
    <pos>n</pos>
  </gramGrp>
  <sense n='1'>
    <gramGrp>
      <number>sometimes sing</number>
    </gramGrp>
    <def>the ability to reason and act, esp. quickly...</def>
    <!-- ... -->
  </sense>
</entry>
```

The following gives the formal declarations for elements in the grammatical-information group.

```
<!-- 12.3.2:  The gram group                                    -->
<!ELEMENT gramGrp       - -  (%m.gramInfo | %paraContent)*      >
<!ATTLIST gramGrp            %a.global;
                             %a.dictionaries;                   >
<!ELEMENT pos           - O  (%paraContent;)                    >
<!ATTLIST pos                %a.global;
                             %a.dictionaries;                   >
<!ELEMENT subc          - O  (%paraContent;)                    >
<!ATTLIST subc               %a.global;
                             %a.dictionaries;                   >
<!ELEMENT colloc        - O  (%paraContent;)                    >
<!ATTLIST colloc             %a.global;
                             %a.dictionaries;
          type               CDATA               #IMPLIED      >
<!-- This fragment is used in sec. 12.1                        -->
```

The class of elements allowed within the <gramGrp> element is declared thus. The class *morphInfo* is defined above in section 12.3.1 ('Information on Written and Spoken Forms') on p. 279.

```
<!-- 12.3.2:  Elements for grammatical information           -->
<!ENTITY % x.gramInfo ''                                      >
<!ENTITY % m.gramInfo '%x.gramInfo colloc | gramGrp | lbl | pos
          | subc | usg'                                       >
<!-- This fragment is used in sec. 12.1                       -->
```

12.3.3 Sense Information

Dictionaries may describe the meanings of words in a wide variety of different ways — by means of synonyms, paraphrases, translations into other languages, formal definitions in various highly stylized forms, etc. No attempt is made here to distinguish all the different forms which sense information may take; all alike may be tagged using the <def> element described in section 12.3.3.1 ('Definitions') on p. 286.

Because as a special case it is frequently desired to distinguish the provision of translation equivalents in other languages from other forms of sense information, however, the specialized elements <tr> (translation equivalent) and <trans> (which groups a translation equivalent with related information such as its grammatical description) are defined for this purpose in section 12.3.3.2 ('Translation Equivalents') on p. 287.

Whether sense information in multilingual dictionaries is consistently tagged using <tr> or <def> is a matter of the encoder's choice; no blanket recommendation is made here.

12.3.3.1 Definitions

Dictionary definitions are those pieces of prose in a dictionary entry that describe the meaning of some lexical item. Most often, definitions describe the headword of the entry; in some cases, they describe translated texts, examples, etc.; see <tr>, section 12.3.3.2 ('Translation Equivalents') on p. 287, and <eg>, section 12.3.5.1 ('Examples') on p. 290. The <def> element directly contains the text of the definition; unlike <form> and <gramGrp>, that is, it does not serve solely to group a set of smaller elements. The close analysis of definition text, such as the tagging of hypernyms, typical objects, etc., is not covered by these Guidelines.

Definitions may occur directly within an entry; when multiple definitions are given, they typically are identified as belonging to distinct senses, as here:

" **demigod** (...) n. 1.a. a being who is part mortal, part god. b. a lesser deity. 2. a godlike person. [CP] "

```
<entry>
<form>
   <orth>demigod</orth>
   <pron> ...</pron></form>
<gramGrp><pos>n</pos></gramGrp>
<sense n='1'>
   <sense n='a'>
      <def>a being who is part mortal, part god.</def>
   </sense>
   <sense n='b'>
      <def>a lesser deity.</def>
   </sense>
</sense>
<sense n='2'>
   <def>a godlike person.</def>
</sense>
</entry>
```

In multilingual dictionaries, it is sometimes possible to distinguish translation equivalents from definitions proper; here a <def> element is distinguished from the translation information within which it appears.

" **rémoulade** [Remulad] nf remoulade, rémoulade (*dressing containing mustard and herbs*). [C/R]"

```
<entry>
  <form>
```

```
    <orth>r&eacute;moulade</orth>
    <pron>Remulad</pron>
  </form>
  <gramGrp>
    <pos>n</pos>
    <gen>f</gen>
  </gramGrp>
  <trans>
    <tr>remoulade</tr>
    <tr>r&eacute;moulade</tr>
    <def>dressing containing mustard and herbs</def>
  </trans>
</entry>
```

The following gives the formal definition of <def>:

```
<!-- 12.3.3.1:  Definition text                              -->
<!ELEMENT def           - O  (%paraContent;)                  >
<!ATTLIST def                %a.global;
                             %a.dictionaries;                 >
<!-- This fragment is used in sec. 12.1                      -->
```

12.3.3.2 Translation Equivalents

Multi-lingual dictionaries contain information about translations of a given word in some source language for one or more target languages. Minimally, the dictionary provides the corresponding translation in the target language; other information, such as morphological information (gender, case), various kinds of usage restrictions, etc., may also be given. If translation equivalents are to be distinguished from other kinds of sense information, they may be encoded using the <tr> element.

As in monolingual dictionaries, the <sense> element is used in multi-lingual dictionaries to group information (forms, grammatical information, usage, translation(s), etc.) about a given sense of a word where necessary, as in monolingual dictionaries. Information about the individual translation equivalents within a sense is grouped using <trans> element. This information may include the translation text (tagged <tr> or <def>), morphological information (<gen>, <case>, etc.), usage notes (<usg>), translation labels (<lbl>), and definitions (<def>).

<trans> contains translation text and related information (within an entry in a multilingual dictionary).

<tr> contains a translation of the headword or an example.

<lbl> in dictionaries, contains a label for a form, example, translation, or other piece of information, e.g. “abbreviation for”, “contraction of”, “literally”, “approximately”, “synonyms:”, etc.

Note how in the following example, different translation equivalents are grouped into the same or different senses, following the punctuation of the source and the usage labels:

“ **dresser** ... (a) (Theat) habilleur m, -euse f; (Comm: window) étalagiste mf. she’s a stylish elle s’habille avec chic; V hair. (b) (tool) (for wood) raboteuse f; (for stone) rabotin m. [C/R] ”

```
<entry n='1'>
  <form><orth>dresser</orth><!-- ... --></form>
  <gramGrp> <pos>n</pos> </gramGrp>
  <sense n='a'>
    <sense>
      <usg type=dom>Theat</usg>
      <trans>
        <tr>habilleur</tr>
        <gen>m</gen>
      </trans>
      <trans>
        <tr>-euse</tr>
        <gen>f</gen>
```

```
      </trans>
    </sense>
    <sense>
      <usg type=dom>Comm</usg>
      <form type=compound>
        <orth>window <oRef></orth>
      </form>
      <trans>
        <tr>etalagiste</tr>
        <gen>mf</gen>
      </trans>
    </sense>
    <eg>
      <q>she's a stylish <oRef></q>
      <trans><tr>elle s'habille avec chic</tr></trans>
    </eg>
    <xr type=see>V. <ref target=hair>hair</ref></xr>
  </sense>
  <sense n='b'>
    <usg type=category>tool</usg>
    <sense>
      <usg type=hint>for wood</usg>
      <trans>
        <tr>raboteuse</tr>
        <gen>f</gen>
      </trans>
    </sense>
    <sense>
      <usg type=hint>for stone</usg>
      <trans>
        <tr>rabotin</tr>
        <gen>m</gen>
    </trans>
    </sense>
  </sense>
</entry>
```

In this encoding, a distinction is made between the translation equivalent ("OAS") and a descriptive phrase providing further information for the user of the dictionary.

" **O.A.S.** ... nf (abrév de **Organisation de l'Armée secrète**) OAS (*illegal military organization supporting French rule of Algeria*). [C/R] "

```
<entry>
  <!-- ... -->
  <trans>
    <tr>OAS</tr>
    <def>illegal military organization supporting French
         rule of Algeria</def>
  </trans>
</entry>
```

Note that <tr> may also be used in monolingual dictionaries when a translation is given for a foreign word:

" **havdalah** or **havdoloh** Hebrew. (Hebrew hAvdA"lA; Yiddish hAv"dOl@) n. Judaism. the ceremony marking the end of the sabbath or of a festival, including the blessings over wine, candles and spices. [literally: separation] [CED] "

```
<entry type=foreign>
  <form>
    <orth>havdalah</orth>
    <orth>havdoloh</orth>
  </form>
  <!-- ... -->
```

```
  <usg type=dom>Judaism</usg>
  <def>the ceremony marking the end of the sabbath or of a festival,
       including the blessings over wine, candles and spices.</def>
  <trans>
    <lbl>literally</lbl>
    <tr>separation</tr>
  </trans>
</entry>
```

The formal definition of these elements is as follows:

```
<!-- 12.3.3.2:  Translation information                    -->
<!ELEMENT trans          - O  (%paraContent |
                               %m.dictionaryParts)*          >
<!ATTLIST trans               %a.global;
                               %a.dictionaries;              >
<!ELEMENT tr             - O  (%paraContent;)               >
<!ATTLIST tr                  %a.global;
                               %a.dictionaries;              >
<!-- This fragment is used in sec. 12.1                     -->
```

12.3.4 Etymological Information

The element <etym> marks a block of etymological information. Etymologies may contain highly structured lists of words in an order indicating their descent from each other, but often also include related words and forms outside the direct line of descent, for comparison. Not infrequently, etymologies include commentary of various sorts, and can grow into short (or long!) essays with prose-like structure. This variation in structure makes it impracticable to define tags which capture the entire intellectual structure of the etymology or record the precise interrelation of all the words mentioned. It is, however, feasible to mark some of the more obvious phrase-level elements frequently found in etymologies, using tags defined in the core tag set or elsewhere in this chapter. Of particular relevance for the markup of etymologies are:

<etym> encloses the etymological information in a dictionary entry.
<lang> name of a language mentioned in etymological or other linguistic discussion.
<date> contains a date in any format.
<mentioned> marks words or phrases mentioned, not used.
<gloss> identifies a phrase or word used to provide a gloss or definition for some other word or phrase.
<pron> contains the pronunciation(s) of the word.
<usg> contains usage information in a dictionary entry.
<lbl> in dictionaries, contains a label for a form, example, translation, or other piece of information, e.g. “abbreviation for”, “contraction of”, “literally”, “approximately”, “synonyms:”, etc.

As in other prose, individual word forms mentioned in an etymological description are tagged with <mentioned> elements. Pronunciations, usage labels, and glosses can be tagged using the <pron>, <usg>, and <gloss> elements defined elsewhere in these Guidelines. In addition, the <lang> element may be used to identify a particular language name where it appears, in addition to using the lang attribute of the <mentioned> element.

Examples:

“ **abismo** m. (del gr. a priv. y byssos, fondo). Sima, gran profundidad. ... ”

```
<entry>
  <form><orth>abismo</orth></form>
  <!-- ... -->
  <etym>del <lang>gr.</> <mentioned>a</> priv. y
       <mentioned>byssos</>, <gloss>fondo</>
  </etym>
  <!-- ... -->
</entry>
```

" **neume** \'n(y)üm\ n [F, fr. ML pneuma, neuma, fr. Gk pneuma breath — more at **pneumatic**]: any of various symbols used in the notation of Gregorian chant ... [WNC]"

```
<entry>
  <!-- ... -->
  <etym><lang>F</> fr. <lang>ML</> <mentioned>pneuma</>
        <mentioned>neuma</> fr. <lang>Gk</>
        <mentioned>pneuma</><gloss>breath</>
        <xr type=etym>more at <ptr target='pneumatic'></xr>
  </etym>
  <def>any of various symbols ...
  <!-- ... -->
</entry>
```

The formal definition for the elements described in this section and not declared elsewhere is:

```
<!-- 12.3.4:  Etymologies                                    -->
<!ELEMENT etym          - O  (%paraContent | usg | lbl | def |
                              trans | tr | (%m.morphInfo) | eg |
                              xr)*                             >
<!ATTLIST etym               %a.global;
                             %a.dictionaries;                 >
<!ELEMENT lang          - -  (%paraContent)                   >
<!ATTLIST lang               %a.global;
                             %a.dictionaries;                 >
<!-- This fragment is used in sec. 12.1                      -->
```

12.3.5 Other Information

12.3.5.1 Examples

Dictionaries typically include examples of word use, usually accompanying definitions or translations. In some cases, the examples are quotations from another source, and are occasionally followed by a citation to the author.

The <eg> element contains usage examples and associated information; the example text itself should be enclosed in a <cit> element, if attributed, or a <q> or <quote> element otherwise. The <cit> element associates a quotation with a bibliographic reference to its source.

<eg> (in a dictionary) contains an example text containing at least one occurrence of the word form, used in the sense being described; examples may be quoted from (named) authors or contrived.

<q> contains a quotation or apparent quotation — a representation of speech or thought marked as being quoted from someone else (whether in fact quoted or not); in narrative, the words are usually those of of a character or speaker; in dictionaries, <q> may be used to mark real or contrived examples of usage.

<quote> contains a phrase or passage attributed by the narrator or author to some agency external to the text.

<cit> A quotation from some other document, together with a bibliographic reference to its source.

Examples frequently abbreviate the headword, and so their transcription will frequently make use of the <oRef> or <oVar> elements described below in section 12.4 ('Headword and Pronunciation References') on p. 297.

Examples:

" **multiplex** /.../ adj tech having many parts: the multiplex eye of the fly. [LDOCE]"

```
<eg>
  <q>the multiplex eye of the fly.</q>
</eg>
```

As the following example shows, <eg> can also contain elements such as <pron>, <def>, etc.

" **some** ... 4. (*S* and *any* are used with *more*): Give me more /s@'mO:(r)/ [OALD] "

```
<sense n='4'>
  <usg type=colloc><oRef type=cap> and <mentioned>any</>
       are used with <mentioned>more</></usg>
  <eg>
    <q>Give me <oRef> more</q>
    <pron extent=part>s@'mO:(r)</pron>
  </eg>
</sense>
```

In multilingual dictionaries, examples may also be accompanied by translations:
" **horrifier** ... vt to horrify. **elle était horrifiée par la dépense** she was horrified at the expense. [C/R] "

```
<entry>
  <!-- ... -->
  <trans><tr>to horrify</tr></trans>
  <eg><q>elle &eacute;tait horrifi&eacute;e par la
         d&eacute;pense</q>
      <trans>
        <tr>she was horrified at the expense.</tr>
      </trans>
  </eg>
</entry>
```

When a source is indicated, the example should be marked with a <cit> element:
" **valeur** ... n. f. ... 2. Vx. Vaillance, bravoure (spécial., au combat). "La valeur n'attend pas le nombre des annees" (Corneille). ... [DNT]"

```
  <sense n='2'>
    <usg type=time>Vx.</usg>
    <def>Vaillance, bravoure (sp&eacute;cial., au combat)</def>
      <eg><cit>
            <q>La valeur n'attend pas le nombre des annees</q>
            <bibl><author>Corneille</author></bibl>
          </cit>
      </eg>
```

The formal definition of <eg> is:

```
<!-- 12.3.5.1:  Examples and citations                        -->
<!ELEMENT eg            - O  (q | quote | cit)+
                                              +(%m.dictionaryParts
                                              |
                                              %m.formPointers)
                                                        >
<!ATTLIST eg                %a.global;
                            %a.dictionaries;              >
<!-- This fragment is used in sec. 12.1                  -->
```

12.3.5.2 Usage Information and Other Labels

Most dictionaries provide restrictive labels and phrases indicating the usage of given words or particular senses. Other labels, not necessarily related to usage, may be attached to forms, translations, cross references, and examples. Usage and other labels should be marked with the following elements:

<usg> contains usage information in a dictionary entry.
<lbl> in dictionaries, contains a label for a form, example, translation, or other piece of information, e.g. "abbreviation for", "contraction of", "literally", "approximately", "synonyms:", etc.

Typical usage labels mark

- temporal use (archaic, obsolete, etc.)
- register (slang, formal, taboo, ironic, facetious, etc.)

- style (literal, figurative, etc.)
- connotative effect (e.g. derogatory, offensive)
- subject field (Astronomy, Philosophy, etc.)
- national or regional use (Australian, U.S., Midland dialect, etc.)

Many dictionaries provide an explanation and/or a list of such usage labels in a preface or appendix. The type of the usage information may be indicated in the `type` attribute on the `<usg>` element. Some typical values are:

`geo` geographic area
`time` temporal, historical era ("archaic", "old", etc.)
`dom` domain
`reg` register
`style` style (figurative, literal, etc.)
`plev` preference level ("chiefly", "usually", etc.)
`acc` acceptability
`lang` language for foreign words, spellings pronunciations, etc.
`gram` grammatical usage

In addition to this kind of information, multilingual dictionaries often provide "semantic cues" to help the user determine the right sense of a word in the source language (and hence the correct translation). These include synonyms, concept subdivisions, typical subjects and objects, typical verb complements, etc. These labels are also marked with the `<usg>` element; sample values for the `type` attribute in these cases include:

`syn` synonym given to show use
`hyper` hypernym given to show usage
`colloc` collocation given to show usage
`comp` typical complement
`obj` typical object
`subj` typical subject
`verb` typical verb
`hint` unclassifiable piece of information to guide sense choice

In this entry, one spelling is marked as geographically restricted:
" **colour** or U.S. **color** ... [CED] "

```
<form>
   <orth>colour</orth>
</form>
<form>
   <usg type=geo>U.S.<usg>
   <orth>color</orth>
</form>
```

In this example, usage labels are used to indicate domains, register, and synonyms associated with different senses:
" **palette** [palEt] nf (a) (Peinture: lit, fig) palette. (b) (Boucherie) shoulder. (c) (aube de roue) paddle; (battoir à linge) beetle; (Manutention, Constr) pallet. [C/R] "

```
<!-- ... -->
  <sense n='a'>
    <usg type=dom>Peinture</usg>
    <usg type=style>lit</usg>
    <usg type=style>fig</usg>
    <trans><tr>palette</tr></trans>
  </sense>
  <sense n='b'>
    <usg type=dom>Boucherie</usg>
    <trans><tr>shoulder</tr></trans>
  </sense>
  <sense n='c'>
    <sense>
```

```
      <usg type=syn>aube de roue</usg>
      <trans><tr>paddle</tr></trans>
    </sense>
    <sense>
      <usg type=syn>battoir &agrave; linge</usg>
      <trans><tr>beetle</tr></trans>
    </sense>
    <sense>
      <usg type=dom>Manutention</usg>
      <usg type=dom>Constr</usg>
      <trans><tr>beetle</tr></trans>
    </sense>
  </sense>
<!-- ... -->
```

When the usage label is hard to classify, it may be described as a "hint":
" **rempaillage** [...] nm reseating, rebottoming (*with straw*). [C/R] "

```
<entry>
  <!-- ... -->
  <trans>
    <tr>reseating</tr>
    <tr>rebottoming</tr>
    <usg type=hint>with straw</usg>
  </trans>
</entry>
```

The following gives the formal definition of <usg> and <lbl>:

```
<!-- 12.3.5.2:  Usage information                              -->
<!ELEMENT usg           - O  (%paraContent;)                    >
<!ATTLIST usg                %a.global;
                             %a.dictionaries;
          type               CDATA               #IMPLIED      >
<!ELEMENT lbl           - O  (%paraContent;)                    >
<!ATTLIST lbl                %a.global;
                             %a.dictionaries;
          type               CDATA               #IMPLIED      >
<!-- This fragment is used in sec. 12.1                        -->
```

12.3.5.3 Cross References to Other Entries

Dictionary entries frequently refer to information in other entries, often using extremely dense notations to convey the headword of the entry to be sought, the particular part of the entry being referred to, and the nature of the information to be sought there (synonyms, antonyms, usage notes, etymology, an illustration, etc.)

Cross references may be tagged in dictionaries using the simple <ref> and <ptr> elements defined in the core tag set (section 6.6 ('Simple Links and Cross References') on p. 147), or the "extended" pointing elements <xref> and <xptr> defined in the additional tag set for linking, segmentation, and alignment (section 14.2 ('Extended Pointers') on p. 340). In addition, the <xr> element may be used to group all the information relating to a cross reference. The following elements may be used for tagging cross references within dictionaries:

<xr> contains a phrase, sentence, or icon referring the reader to some other location in this or another text.

<ref> defines a reference to another location in the current document, in terms of one or more identifiable elements, possibly modified by additional text or comment.

<ptr> defines a pointer to another location in the current document in terms of one or more identifiable elements.

<xref> defines a reference to another location in the current document, or an external document, using an extended pointer notation, possibly modified by additional text or comment.

<xptr> defines a pointer to another location in the current document or an external document.

<lbl> in dictionaries, contains a label for a form, example, translation, or other piece of information, e.g. "abbreviation for", "contraction of", "literally", "approximately", "synonyms:", etc.

As in other types of text, the actual pointing element (e.g. <ref> or <ptr>) is used to tag the cross-reference target proper (in dictionaries, usually the headword, possibly accompanyied by a homograph number, a sense number, or other further restriction specifiying what portion of the target entry is being referred to); as usual, a <ptr> element may be used when the actual text of the target description can be reconstructed automatically, and a <ref> element is typically to be preferred when it cannot be reconstructed automatically.

The <xr> element is used to group the target with any accompanying phrases or symbols used to label the cross reference; the cross reference label itself may be tagged as a <lbl> or may remain untagged. Both of the following are thus legitimate:

" **glee** ... Compare **madrigal** (sense 1) [CED] "

```
<entry><form><orth>glee</></>
  <!-- ... -->
  <xr>Compare <ptr target='madrigal.1'></xr>
</entry>
```

" **hostellerie** Syn. de hôtellerie (sens 1). [DNT] "

```
<xr type=syn>
  <lbl>Syn. de</lbl> <ref>h&ocirc;tellerie (sens 1)</ref>.
</xr>
```

In addition to using, or not using, <lbl> to mark the cross-reference label, the two examples differ in another way. The former assumes that the first sense of 'madrigal' has the SGML identifier "madrigal.1", and that the specific form of the reference in the source volume can be reconstructed, if needed, from that information. The latter does not require the first sense of "hôtellerie" to have an SGML identifier, and retains the print form of the cross reference; by omitting the target attribute of the <ref> element, however, the second example does assume implicitly either that some software could usefully parse the phrase tagged as a <ref> and find the location referred to, or else that such processing will not be necessary.

The type attribute on the pointing element or on the <xr> element may be used to indicate what kind of cross reference is being made, using any convenient typology. Since different dictionaries may label the same kind of cross reference in different ways, it may be useful to give normalized indications in the type attribute, enabling the encoder to distinguish irregular forms of cross reference more reliably:

" **rose**2 ... vb. the past tense of **rise**. [CED] "

```
<entry type='xr, hom' n=2>
  <form><orth>rose</orth><!-- ... --></form>
  <!-- ... -->
  <xr type=pastof>
    <lbl>the past tense of</lbl>
    <ref target='rise'>rise</ref>
  </xr>
</entry>
```

from cross-references for synonyms and the like:

" **antagonist** ... syn see **adverse** [W7] "

```
<xr type=syn>
  <lbl>syn see</lbl>
  <ref target='adverse'>adverse</ref>
</xr>
```

Strictly speaking, this reference is not to the entry given, but to the list of synonyms it contains. Slightly more complicated is this reference to an illustration accompanying another article:

" **ax, axe** ... ⟶ see the illus at **tool** [OALD] "

This entry refers to the illustration at the entry for 'tool', not the entry itself. The target attribute might give the identifier of the illustration itself, or of the enclosing entry (in which case

the type attribute might be used to infer that the reference is actually to the illustration, not the entry as a whole).

```
<xr type='see illustration'>
  <lbl>see the illus at</lbl>
  <ptr target='tool.illus'>
</xr>
```

In some cases, the cross reference is to a particular subset of the meanings of the entry in question:

" **globe** ...V. **armillaire** (sphère) [PR] "

```
<xr>V. <ref target='armillaire'>armillaire</ref>
       <lbl type='sense-restriction'>sph&egrave;re</lbl>
</xr>
```

Cross-references occasionally occur in definition texts, example texts, etc., or may be free-standing within an entry. These may typically be encoded using <ref> or <ptr>, without an enclosing <xr>. For example:

" **entacher** ... *Acte entaché de nullité,* contenant un vice de forme ou passé par un incapable*. [DNT] "

The asterisk signals a reference to the entry for 'incapable'.

```
<def>contenant un vice de forme ou pass&eacute;
par un <ptr target='incapable'>.</def>
```

In some cases, the form in the definition is inflected, and thus <ref> must be used, as here:

" **justifier** ...4. IMPRIM Donner a (une ligne) une longeur convenable au moyen de blancs (2, sens 1, 3). [DNT] "

```
<sense n='4'>
   <usg type=dom>imprim</usg>
   <def>Donner a (une ligne) une longeur convenable au moyen
        de <ref target='blanc-2.1 blanc-2.3'>blancs (2, sens
        1, 3)</ref>
   </def>
</sense>
```

The formal definition for <xr> is as follows:

```
<!-- 12.3.5.3:  Cross References                              -->
<!ELEMENT xr            - O  (%paraContent | usg | lbl)*       >
<!ATTLIST xr                 %a.global;
                             %a.dictionaries;
          type               CDATA               #IMPLIED      >
<!-- This fragment is used in sec. 12.1                       -->
```

12.3.5.4 Notes within Entries

Dictionaries may include extensive explanatory notes about usage, grammar, context, etc. within entries. Very often, such notes appear as a separate section at the end of an entry. The <note> element should be used for such material.

<note> contains a note or annotation.

For example:

"

ain't (eInt) *Not standard. contraction of* am not, is not, are not, have not *or* has not: *I ain't seen it.* ....

▲**Usage.** Although the interrogative form *ain't I?* would be a natural contraction of *am I not?*, it is generally avoided in spoken English and never used in formal English. [CED] "

```
<entry>
  <form type=contr>
    <orth>ain't</orth>
    <pron>eInt</pron>
  </form>
```

```
  <usg type=reg>Not standard</usg>
  <form type=full>
    <lbl>contraction of</lbl>
    <orth>am not</orth>
    <orth>is not</orth>
    <orth>are not</orth>
    <orth>have not</orth>
    <orth>has not</orth>
  </form>
  <eg>
    <q>I ain't seen it.</q>
  </eg><!-- ... -->
  <note type=usage>
     Although the interrogative form <mentioned>ain't
     I?</> would be a natural contraction of <mentioned>am
     I not?</>, it is generally avoided in spoken English
     and never used in formal English.
  </note>
</entry>
```

The formal declaration for <note> is given in section 6.8 ('Notes, Annotation, and Indexing') on p. 152. It has this form:

```
<!-- 12.3.5.4:  [Note]                                        -->
<!ELEMENT note          - O  (%specialPara;)                    >
<!ATTLIST note
          id                ID                  #IMPLIED
          lang              IDREF               %INHERITED
          rend              CDATA               #IMPLIED
          n                 CDATA               #IMPLIED
          type              CDATA               #IMPLIED
          resp              CDATA               #IMPLIED
          place             CDATA               'unspecified'
          anchored          (yes | no)          yes
          target            IDREFS              #IMPLIED
          targetEnd         IDREFS              #IMPLIED       >
```

12.3.6 Related Entries

The <re> element encloses a degenerate entry which appears in the body of another entry for some purpose. Many dictionaries include related entries for direct derivatives or inflected forms of the entry word, or for compound words, phrases, collocations, and idioms containing the entry word.

Related entries can be complex, and may in fact include any of the information to be found in a regular entry. Therefore, the <re> element is defined to contain the same elements as an <entry> element, with the exception that it may not contain any nested <re> elements.

Examples:

" **bevvy** ("bEvI) Dialect. n., pl. -vies. 1. a drink, esp. an alcoholic one: we had a few bevvies last night. 2. a night of drinking. vb. - vies, -vying, -vied (intr.) 3. to drink alcohol [probably from Old French vevee, buvee, drinking] –'bevvied adj. [CED] "

```
<entry>
  <form>
    <orth>bevvy</orth>
    <pron>"bEvI</pron>
  </form>
  <usg type=reg>Dialect</usg>
  <hom>
    <gramGrp><pos>n</pos></gramGrp>
    <!-- ... -->
    <sense n='1'>
      <def>a drink, esp. an alcoholic one</def>
```

```
      <!-- ... -->
    </sense>
    <!-- ... -->
  </hom>
  <hom>
    <gramGrp><pos>vb</pos></gramGrp>
    <!-- ... -->
    <sense n='3'><def>to drink alcohol</def></sense>
  </hom>
  <etym>probably from <lang>Old French</lang>
       <mentioned>bevee</>, <mentioned>buvee</>
       <gloss>drinking</gloss>
  </etym>
  <re type=derived>
    <form>
      <orth>'bevvied</orth>
    </form>
    <gramGrp>
      <pos>adj</pos>
    </gramGrp>
  </re>
</entry>
```

The formal definition of <re> is

```
<!-- 12.3.6:  Related entries                                  -->
<!ELEMENT re            - O  (sense | %m.dictionaryTopLevel |
                              %m.phrase | #PCDATA)*

                                                  -(re)         >
<!ATTLIST re                 %a.global;
                             %a.dictionaries;
          type               CDATA               #IMPLIED      >
<!-- This fragment is used in sec. 12.1                        -->
```

12.4 Headword and Pronunciation References

Examples, definitions, etymologies, and occasionally other elements such as cross references, orthographic forms, etc., often contain a shortened or iconic reference to the headword, rather than repeating the headword itself. The references may be to the orthographic form or to the pronunciation, to the form given or to a variant of that form. The following elements are used to encode such iconic references to a headword:

<oRef> in a dictionary example, indicates a reference to the orthographic form(s) of the headword. Attributes include:

type indicates the kind of typographic modification made to the headword in the reference. Sample values include:

cap indicates first letter is given as capital

nohyph indicates that the headword, though a prefix or suffix, loses its hyphen

<pRef> in a dictionary example, indicates a reference to the pronunciation(s) of the headword.

<oVar> in a dictionary example, indicates a reference to variant orthographic form(s) of the headword. Attributes include:

type indicates the kind of variant involved. Sample values include:

pt past tense

pp past participle

prp present participle

f feminine

pl plural

<pVar> in a dictionary example, indicates a reference to variant pronunciation(s) of the headword.

As members of the class *formPointers*, all these elements share a target attribute, which may optionally be used to resolve any ambiguity about the headword form being referred to.

target gives the SGML identifier of the orthographic form referred to.

Headword references come in a variety of formats:

indicates a reference to the full form of the headword
pref gives a prefix to be affixed to the headword
suf gives a suffix to be affixed to the headword
A gives the first letter in upper case, indicating that the headword is capitalized
pref suf gives a prefix and a suffix to be affixed to the headword
a. gives the initial of the word followed by a full stop, to indicate reference to the full form of the headword
A. refers to a capitalized form of the headword

The <oRef> element should be used for iconic or shortened references to the orthographic form(s) of the headword itself. It is an empty element and replaces, rather than enclosing, the reference. Note that the reference to a headword is not necessarily a simple string replacement. In the example "**colour**l, (US = color) ... films; TV; Red, blue and yellow are s." [OALD], the tilde stands for either headword form ('colour', 'color').

Examples:

" **colonel** ... army officer above a lieutenant- . [OALD] "

```
<def>army officer above a lieutenant-<oRef></def>
```

"**academy** ... The Royal A of Arts [OALD]"

```
<q>The Royal <oRef type=cap> of Arts</q>
```

The following example demonstrates the use of the target attribute to refer to a specific form of the headword:

" **vag-** or **vago-** comb form ... : vagus nerve < **vag**al > < **vago**tomy > [W7] "

```
<entry>
  <form>
    <orth id=o1>vag-</orth>
    <orth id=o2>vago-</orth>
  </form>
  <!-- ... -->
  <def>vagus nerve</def>
  <eg>
    <q><oRef target=o1 type=nohyph>al</q>
    <q><oRef target=o2 type=nohyph>tomy</q>
  </eg>
</entry>
```

In many cases the reference is not to the orthographic form of the headword, but rather to another form of the headword — usually to an inflected form. In these cases, the element <oVar> should be used; this element takes as its content the string as it appears in the text.

" **take** ... < Mr Burton **took** us for French > [NPEG] "

```
<eg>
  <q>Mr Burton <oVar type=pt>took</oVar> us for French</q>
</eg>
```

" **take** ... < was quite **n** with him > [NPEG] "

```
<eg>
  <q>was quite <oVar type=pp><oRef>n</oVar> with him</q>
</eg>
```

Note that, in the example above, tilde and font shift combine. The encoding

```
<q>was quite <oRef>n with him</q>
```

would correspond to "was quite n with him" (no font shift).

The next example shows a discontinuous reference, using the attributes next and prev, which are defined in the additional tag set for linking, segmentation, and alignment (see chapter 14 ('Linking, Segmentation, and Alignment') on p. 331) and therefore require that that tag set be selected in addition to that for dictionaries.

" **mix up**... < it's easy to **mix** her **up** with her sister > [NPEG] "

```
<eg><q>it's easy to <oVar next=ov2 id=ov1>mix</oVar>
     her <oVar prev=ov1 id=ov2>up</oVar> with her
     sister</q>
</eg>
```

In addition, some dictionaries make reference to the pronunciation of the headword in the pronunciation of related entries, variants, or examples. The <pRef> and <pVar> elements should be used for such references.

" **hors d'oeuvre** /,aw'duhv (Fr O:r dœvr)/ n, pl hors d'oeuvres also hors d'oeuvre /'duhv(z) (Fr)/ [NPEG] "

```
<form>
  <orth>hors d'oeuvre</orth>
  <pron>%aU"dUv</pron>
  <form>
    <usg type=lang>Fr</usg>
    <pron id=p2>OR dOvR</pron>
  </form>
</form>
<!-- ... -->
<form type=infl>
  <number>pl</number>
  <orth>hors d'oeuvres</orth>
  <orth>hors d'oeuvre</orth>
  <pron extent=part>"dUv(z)</pron>
  <form>
    <usg type=lang>Fr</usg>
    <pron><pRef target=p2></pron>
  </form>
</form>
```

Because headword and pronunciaton references can occur virtually anywhere in an entry, the <oRef>, <oVar>, <pRef>, and <pVar> elements can appear within any other element defined for dictionary entries.

Since existing printed dictionaries use different conventions for headword references (swung dash, first letter abbreviated form, capitalization or italicization of the word, etc.) the exact method used should be documented in the header.

The class of headword references is defined thus:

```
<!-- 12.4:  Classes for headword references                  -->
<!ENTITY % x.formPointers ''                                   >
<!ENTITY % m.formPointers '%x.formPointers oRef | oVar | pRef |
           pVar'                                               >
<!ENTITY % a.formPointers '
          target             IDREF               #IMPLIED'     >
<!-- This fragment is used in sec. 12.1                        -->
```

The formal declaration for headword reference elements is:

```
<!-- 12.4:  Headword references                                -->
<!ELEMENT oRef          - O  EMPTY                             >
<!ATTLIST oRef               %a.global;
                             %a.dictionaries;
                             %a.formPointers;
          type               CDATA               #IMPLIED      >
```

```
<!ELEMENT oVar          - -  (#PCDATA | oRef)*                  >
<!ATTLIST oVar               %a.global;
                             %a.dictionaries;
                             %a.formPointers;
          type               CDATA                #IMPLIED    >
<!ELEMENT pRef          - O  EMPTY                              >
<!ATTLIST pRef               %a.global;
                             %a.dictionaries;
                             %a.formPointers;                   >
<!ELEMENT pVar          - -  (#PCDATA | pRef)*                  >
<!ATTLIST pVar               %a.global;
                             %a.dictionaries;
                             %a.formPointers;                   >
<!-- This fragment is used in sec. 12.1                       -->
```

12.5 Typographic and Lexical Information in Dictionary Data

Among the many possible views of dictionaries, it is useful to distinguish at least the following three, which help to clarify some issues raised with particular urgency by dictionaries, on account of the complexity of both their typography and their information structure.

- (a) the *typographic view*, which is concerned with the two-dimensional printed page, including information about line and page breaks and other features of layout
- (b) the *editorial view* — the one-dimensional sequence of tokens which can be seen as the input to the typesetting process; the wording and punctuation of the text and the sequencing of items are visible in this view, but specifics of the typographic realization are not
- (c) the *lexical view* — this view includes the underlying information represented in a dictionary, without concern for its exact textual form

For example, a domain indication in a dictionary entry might be broken over a line and therefore hyphenated ("naut-" "ical"); the typographic view of the dictionary preserves this information. In a purely editorial view, the particular form in which the domain name is given in the particular dictionary (as "nautical", rather than "naut.", "Naut.", etc.) would be preserved, but the fact of the line break would not. Font shifts might plausibly be included in either a strictly typographic or an editorial view. In the lexical view, the only information preserved concerning domain would be some standard symbol or string representing the nautical domain (e.g. "naut.") regardless of the form in which it appears in the printed dictionary.

In practice, publishers begin with the lexical view — i.e., lexical data as it might appear in a database — and generate first the editorial view, which reflects editorial choices for a particular dictionary (such as the use of the abbreviation "Naut." for "nautical", the fonts in which different types of information are to be rendered, etc.), and then the typographic view, which is tied to a specific printed rendering. Computational linguists and philologists often begin with the typographic view and analyse it to obtain the editorial and/or lexical views. Some users may ultimately be concerned with retaining only the lexical view, or they may wish to preserve the typographic or editorial views as a reference text, perhaps as a guard against the loss or misinterpretation of information in the translation process. Some researchers may wish to retain all three views, and study their interrelations, since research questions may well span all three views.

In general, an electronic encoding of a text will allow the recovery of at least one view of that text (the one which guided the encoding); if editorial and typographic practices are consistently applied in the production of a printed dictionary, or if exceptions to the rules are consistently recorded in the electronic encoding, then it is *in principle* possible to recover the editorial view from an encoding of the lexical view, and the typographic view from an encoding of the editorial view. In practice, of course, the severe compression of information in dictionaries, the variety of methods by which this compression is achieved, the complexity of formulating completely explicit

rules for editorial and typographic practice, and the relative rarity of complete consistency in the application of such rules, all make the mechanical transformation of information from one view into another something of a vexed question.

This section describes some principles which may be useful in capturing one or the other of these views as consistently and completely as possible, and describes some methods of attempting to capture more than one view in a single encoding. Only the editorial and lexical views are explicitly treated here; for methods of recording the physical or typographic details of a text, see chapter 18 ('Transcription of Primary Sources') on p. 443. Other approaches to these problems, such as the use of repetitive encoding and links to show their correspondences, or the use of feature structures to capture the information structure, and of the `ana` and `inst` attributes to link feature structures to a transcription of the editorial view of a dictionary, are not discussed here. (For feature structures, see chapter 16 ('Feature Structures') on p. 397. For linkage of textual form and underlying information, see chapter 15 ('Simple Analytic Mechanisms') on p. 381.)

12.5.1 Editorial View

Common practice in encoding texts of all sorts relies on principles such as the following, which can be used successfully to capture the editorial view when encoding a dictionary:

1. All characters of the source text should be retained, with the possible exception of *rendition text* (for which see further below).
2. Characters appearing in the source text should typically be given as character data content in the SGML document, rather than as the value of an attribute; again, rendition text may optionally be excepted from this rule.
3. Apart from the characters or graphics in the source text, nothing else should appear as content in the SGML document, although it may be given in attribute values.
4. The material in the source text should appear in the encoding in the same order. Complications of the character sequence by footnotes, marginal notes, etc., text wrapping around illustrations, etc., may be dealt with by the usual means (for notes, see section 6.8 ('Notes, Annotation, and Indexing') on p. 152).[4]

In a very conservative transcription of the editorial view of a text, *rendition characters* (e.g. the commas, parentheses, etc., used in dictionary entries to signal boundaries among parts of the entry) and *rendition text* (for example, conjunctions joining alternate headwords, etc.) are typically retained. Removing the SGML tags from such a transcription will leave all and only the characters of the source text, in their original sequence.[5]

Consider, for example, the following entry:

"**pinna** ('pIn@) n., pl. -nae (-ni:) *or* -nas. 1. any leaflet of a pinnate compound leaf. 2. *Zoology*. a feather, wing, fin, or similarly shaped part. 3. another name for **auricle** (sense 2). [C18: via New Latin from Latin: wing, feather, fin] [CED]"

A conservative encoding of the editorial view of this entry, which retains all rendition text, might resemble the following:

```
<entry>
  <form>
    <orth>pinna</orth>
    <pron>("pIn@)</pron>
  </form>
  <gramGrp><pos>n.</pos>, </gramGrp>
  <form type=infl>
    <number>pl.</number>
```

[4]Complications of sequence caused by marginal or interlinear insertions and deletions, which are frequent in manuscripts, or by unconventional page layouts, as in concrete poetry, magazines with imaginative graphic designers, and texts about the nature of typography as a medium, typically do not occur in dictionaries, and so are not discussed here.

[5]This is a slight oversimplification. Even in conservative transcriptions, it is common to omit page numbers, signatures of gatherings, running titles and the like. The simple description above also elides, for the sake of simplicity, the difficulties of assigning a meaning to the phrase "original sequence" when it is applied to the printed characters of a source text; the "original sequence" retained or recovered from a conservative transcription of the editorial view is, of course, the one established during the transcription by the encoder.

```
    <form>
      <orth type=lat extent=part>-nae</orth>
      <pron extent=part>(-ni:)</pron>
    </form>
    or
    <orth type=std extent=part>-nas</orth>
  </form>
  <sense n='1'>
    1. <def>any leaflet of a pinnate compound leaf.</def>
  </sense>
  <sense n='2'>
    2. <usg type=dom>Zoology</usg>
    <def>a feather, wing, fin, or similarly shaped part.</def>
  </sense>
  <sense n='3'>
    3.<xr type=syn>
      <lbl>another name for</lbl>
      <ref target='auricle.2'>auricle (sense 2).</ref>
    </xr>
  </sense>
  <etym>
     [<date>C18</date>: via <lang>New Latin</lang> from
     <lang>Latin</lang>: <gloss>wing</gloss>, <gloss>feather</gloss>,
     <gloss>fin</gloss>]
  </etym>
</entry>
```

A somewhat simplified encoding of the editorial view of this entry might exploit the fact that rendition text is often systematically recoverable. For example, parentheses consistently appear around pronunciation in this dictionary, and thus are effectively implied by the start- and end-tags for <pron>.[6] In such an encoding, removing the tags should exactly reproduce the sequence of characters in the source, minus rendition text. The original character sequence can be recovered fully by replacing tags with any rendition text they imply.

Encoding in this way, the example given above might resemble the following. The <tagUsage> element in the header would be used to record the following patterns of rendition text:

- parentheses appear around <pron> elements
- commas appear before inflected forms
- the word "or" appears before alternate forms
- brackets appear around the etymology
- full stops appear after <pos>, inflection information, and sense numbers
- senses are numbered in sequence unless otherwise specified using the global n attribute

```
<entry>
  <form>
    <orth>pinna</orth>
    <pron>"pIn@</pron>
  </form>
  <gramGrp><pos>n</pos></gramGrp>
  <form type=infl>
    <number>pl</number>
    <form>
      <orth type=lat extent=part>-nae</orth>
      <pron extent=part>-ni:</pron>
    </form>
    <orth type=std extent=part>-nas</orth>
  </form>
```

[6] The omission of rendition text is particularly common in systems for document production; it is considered good practice there, since automatic generation of rendition text is more reliable and more consistent than attempting to maintain it manually in the electronic text.

```
    <sense n='1'>
      <def>any leaflet of a pinnate compound leaf.</def>
    </sense>
    <sense n='2'>
      <usg type=dom>Zoology</usg>
      <def>a feather, wing, fin, or similarly shaped part.</def>
    </sense>
    <sense n='3'>
      <xr type=syn>
        <lbl>another name for</lbl>
        <ref>auricle (sense 2).</ref>
      </xr>
    </sense>
    <etym>
       <date>C18</date>: via <lang>New Latin</lang> from
       <lang>Latin</lang>: <gloss>wing</gloss>, <gloss>feather</gloss>,
       <gloss>fin</gloss>
    </etym>
  </entry>
```

When rendition text is omitted, it is recommended that the means to regenerate it be fully documented, using the <tagUsage> element of the TEI header.

If rendition text is used systematically in a dictionary, with only a few mistakes or exceptions, the global attribute rend may be used on any tag to flag exceptions to the normal treatment. The values of the rend attribute are not prescribed, but it can be used with values such as no-comma, no-left-paren, etc. Specific values can be documented using the <rendition> element in the TEI header.

In the following (imaginary) example, no left parenthesis precedes the pronunciation:
"**biryani** or **biriani** %bIrI"A:nI) any of a variety of Indian dishes ... [from Urdu]"
This irregularity can be recorded thus:

```
<entry>
  <form>
    <orth>biryani</orth>
    <orth>biriani</orth>
    <pron rend=noleftparen>%bIrI"A:nI</pron>
  </form>
  <def>any of a variety of Indian dishes ... </def>
  <etym>from <lang>Urdu</lang></etym>
</entry>
```

12.5.2 Lexical View

If the text to be interchanged retains only the lexical view of the text, there may be no concern for the recoverability of the editorial (not to speak of the typographic) view of the text. However, it is strongly recomended that the TEI header be used to document fully the nature of all alterations to the original data, such as normalization of domain names, expansion of inflected forms, etc.

In an encoding of the lexical view of a text, there are degrees of departure from the original data: normalizing inconsistent forms like "nautical", "naut"., "Naut.", etc., to "nautical" is a relatively slight alteration; expansion of "delay -ed -ing" to "delay, delayed, delaying" is a more substantial departure. Still more severe is the rearranging of the order of information in entries — for example,

- reorganizing the order of elements in an entry to show their relationship, as in "**clem** (klEm) or clam vb. clems, clemming, clemmed or clams, clammimg, clammed [CED]" where in a strictly lexical view it would be necessary to group "clem" and "clam" with their respective inflected forms.
- splitting an entry into two separate entries, as in "**celi.bacy** /"selIb@sI/ n [U] state of living unmmarried, esp as a religious obligation. celi.bate /"selIb@t/ n [C] unmmarried person (esp a priest who has taken a vow not to marry). [OALD]"

For some purposes, this entry might usefully be split into an entry for "celibacy" and a separate entry for "celibate".

An encoding which captures the lexical view of the example given in the previous section might look something like the following. In this encoding,

- abbreviated forms have been silently expanded
- some forms have been moved to allow related forms to be grouped together
- the part of speech information has been moved to allow all forms to be given together
- the cross reference to "auricle" has been simplified

```
<entry>
  <form>
    <orth>pinna</orth>
    <pron>"pIn@</pron>
    <form type=infl>
      <number>pl</number>
      <form>
        <orth type=lat>pinnae</orth>
        <pron>'pIni:</pron>
      </form>
      <orth type=std>pinnas</orth>
    </form>
  </form>
  <gramGrp>
    <pos>n</pos>
  </gramGrp>
  <sense n='1'>
    <def>any leaflet of a pinnate compound leaf.</def>
  </sense>
  <sense n='2'>
    <usg type=dom>Zoology</usg>
    <def>a feather, wing, fin, or similarly shaped part.</def>
  </sense>
  <sense n='3'>
    <xr type=syn><ptr target=auricle.2></xr>
  </sense>
  <etym>
     <date>C18</date>: via <lang>New Latin</> from
     <lang>Latin</>: <gloss>wing</>, <gloss>feather</>,
     <gloss>fin</>
  </etym>
</entry>
```

12.5.3 Retaining Both Views

It is sometimes desirable to retain both the lexical and the editorial view, in which case a potential conflict exists between the two. When there is a conflict between the encodings for the lexical and editorial views, the principles described in the following sections may be applied.

12.5.3.1 Using Attribute Values to Capture Alternate Views

If the order of the data is the same in both views, then both views may be captured by encoding one "dominant" view in the character data content of the document, and encoding the other using attribute values on the appropriate elements. If all SGML tags were to be removed, the remaining characters would be those of the dominant view of the text.

The attribute class *dictionaries* is used to provide attributes for use in encoding multiple views of the same dictionary entry. These attributes are available for use on all elements when the base tag set for dictionaries is selected.

When the editorial view is dominant, the following attributes may be used to capture the lexical view:

norm gives a normalized form of information given by the source text in a non-normalized form

split gives the list of split values for a merged form

When the lexical view is dominant, the following attributes may be used to record the editorial view:

orig gives the original string or is the empty string when the element does not appear in the source text.

mergedin gives a reference to another element, where the original appears as a merged form.

One attribute is useful in either view:

opt indicates whether the element is optional or not

For example, if the source text had the domain label "naut.", it might be encoded as follows. With the editorial view dominant:

```
<usg type=dom norm='nautical'>naut.</usg>
```

The lexical view of the same label would transcribe the normalized form as content of the <usg> element, the typographic form as an attribute value:

```
<usg type=dom orig='naut.'>nautical</usg>
```

If the source text gives inflectional information for the verb 'delay' as "delay, -ed, -ing", it might usefully be expanded to "delayed, delayed, delaying". An encoding of the editorial view might take this form:

```
<form>
  <orth>delay</orth>
  <form type=infl>
    <orth norm='delayed' extent=part>-ed</orth>
    <tns norm='pst,pstp'></tns>
  </form>
  <form type=infl>
    <orth norm='delaying' extent=part>-ing</orth>
    <tns norm='prsp'></tns>
  </form>
</form>
```

Note the use of the <tns> tag with null content, to enable the representation of implicit information even though it has no print realization.

The lexical view might be encoded thus:

```
<form>
  <orth>delay</orth>
  <form type=infl>
    <orth orig='-ed'>delayed</orth>
    <tns orig=''>pst</tns>
    <tns orig=''>pstp</tns>
  </form>
  <form type=infl>
    <orth orig='-ing'>delaying</orth>
    <tns orig=''>prsp</tns>
  </form>
</form>
```

A particular problem may be posed by the common practice of presenting two alternate forms of a word in a single string, by marking some parts of the word as optional in some forms. The following entry is for a word which can be spelled either "thyrostimuline" or "thyréostimuline":

"**thyr(é)ostimuline** [tiR(e)ostimylin] ..."

With the editorial view dominant, this entry might begin thus:

```
<form>
  <orth split='thyrostimuline,
               thyr&eacute;ostimuline'>
        thyr(&eacute;)ostimuline</orth>
```

```
    <pron split='tiRostimylin,
                 tiReostimylin'>
          tiR(e)ostimylin</pron>
  </form>
```

With the lexical view dominant, however, two <orth> and two <pron> elements would be encoded, in order to disentangle the two forms; the orig attribute would be used to record the typographic presentation of the information in the source.

```
  <form>
    <orth orig='thyr(&eacute;)ostimuline' id=o1>
          thyrostimuline</orth>
    <pron orig='tiR(e)ostimylin' id=p1>
          tiRostimylin</pron>
  </form>
  <form>
    <orth mergedin=o1>thyr&eacute;ostimuline</orth>
    <pron mergedin=p1>tiReostimylin</pron>
  </form>
```

This example might also be encoded using the opt attribute combined with the attributes next and prev defined in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331.

```
  <form>
    <orth next=o2 id=o1>thyr</orth>
    <orth prev=o1 next=o3 opt=y id=o2>&eacute;</orth>
    <orth prev=o2 id=o3>ostimuline</orth>

    <pron next=p2 id=p1>tiR</pron>
    <pron prev=p1 next=p3 opt=y id=p2>e</pron>
    <pron prev=p2 id=p3>ostimylin</pron>
  </form>
```

Note that this transcription preserves both the lexical and editorial views in a single encoding. However, it has the disadvantage that the strings corresponding to entire words do not appear in the encoding uninterrupted, and therefore complex processing is required to retrieve them from the encoded text. The use of the opt attribute is recomended, however, when long spans of text are involved, or when the optional part contains embedded tags.

For example, the following gives two definitions in one text: "picture drawn with coloured chalk made into crayons", and "coloured chalk made into crayons":

"**pas.tel** /"p&stl US: p&"stel/ n 1 (picture drawn with) coloured chalk made into crayons. 2... [OALD]"

A simple encoding solution would be to leave the definition text unanalysed, but this might be felt inadequate since it does not show that there are two definitions. A possible alternative encoding would be:

```
  <sense n='1'>
    <def>coloured chalk made into crayons</def>
    <def>picture drawn with coloured chalk made into crayons</def>
  </sense>
```

This transcribes some characters of the source text twice, however, which deviates from the usual practice. The following encoding records both the editorial and lexical views:

```
  <sense n='1'>
    <def opt=y next=d2 id=d1>picture drawn with</def>
    <def prev=d1 id=d2>coloured chalk made into crayons</def>
  </sense>
```

A more complex example is the following, in which the optional element contains additional tags:

"**canary** ...(Geog) Canary Isles, Canaries (îles fpl) Canaries fpl... [C/R]"

```
  <re type=cmpd>
    <usg type=dom>Geog</usg>
```

```
  <form>
    <orth>Canary Isles</orth>
    <orth>Canaries</orth>
  </form>
  <trans id=t1 opt=y next=t2>
    <tr>&icirc;les</tr>
    <gen>f</gen>
    <number>pl</number>
  </trans>
  <trans id=t2 prev=t1>
    <tr>Canaries</tr>
    <gen>f</gen>
    <number>pl</number>
  </trans>
</re>
```

12.5.3.2 Recording Original Locations of Transposed Elements

The attributes described in the previous section are useful only when the order of material is the same in both the editorial and the lexical view. When the two views impose different orders on the data, the SGML ID/IDREF mechanism may be used to show the original location of material transposed in an encoding of the lexical view.

If the original is only slightly modified, the <anchor> element may be used to mark the original location of the material, and the location attribute may be used on the lexical encoding of that material to indicate its original location(s). Like those in the preceding section, this attribute is defined for the attribute class *dictionaries*:

opt indicates whether the element is optional or not

For example:
" **pinna** ("pIn@) n., pl. -nae (-ni:) or -nas. [CED] "

```
<form>
  <orth>pinna</orth>
  <pron>'pIn@</pron>
  <anchor id=p1>
  <form type=infl>
    <number>pl</number>
    <form>
      <orth extent=part>-nae</orth>
      <pron extent=part>-ni:</pron>
    </form>
    <orth extent=part>-nas</orth>
  </form>
</form>
<gramGrp>
  <pos location=p1>n</pos>          <!-- moved -->
</gramGrp>
```

12.5.4 Attributes for Dictionary Elements

The attributes provided for all dictionary-specific elements and documented in this section are defined thus:

```
<!-- 12.5.4:  Attributes for dictionary work                  -->
<!ENTITY % a.dictionaries '
          expand             CDATA              #IMPLIED
          norm               CDATA              #IMPLIED
          split              CDATA              #IMPLIED
          value              CDATA              #IMPLIED
          orig               CDATA              #IMPLIED
          mergedin           IDREF              #IMPLIED
```

```
          opt                (y | n)            n'             >
<!-- This fragment is used in sec. 12.1                        -->
```

12.6 Unstructured Entries

The content model for the <entry> element provides an entry structure suitable for many average dictionaries, as well as many regular entries in more exotic dictionaries. However, the structure of some dictionaries does not allow the restrictions imposed by the content model for <entry>. To handle these cases, the <entryFree> element is defined, which allows for much wider variation in entry structure. Its content model places no constraints at all upon the entry: any element defined in this chapter, as well as all the phrase-level and inter-level elements defined in the core tag set, can appear anywhere in an entry. With the <entryFree> element, the encoder is free to use any element anywhere, as well as to use or omit grouping elements such as <form>, <gramGrp>, etc.

The <entryFree> element allows the encoding of entries which violate the structure specified for the <entry> element. For example, in the following entry from a dictionary already in electronic form, it is necessary to include a <pron> element within a <def>. This is not permitted in the content model for <entry>, but it poses no problem in the <entryFree> element.

```
<ent h=demigod><hwd>demi|god</hwd><pr><ph>"demIgQd</ph></pr>
<hps ps=n>
<hsn><def>one who is partly divine and partly human</def>
<def>(in Gk myth, etc) the son of a god and a mortal woman,
eg<cf>Hercules</cf><pr><ph>"h3:kjUli:z</ph></pr></def>
</hsn></hps></ent>

[OALD electronic]
```

```
<entryFree>
  <form>
    <orth>demigod</orth>
    <hyph>demi|god</hyph>
    <pron>"demIgQd</pron>
  </form>
  <gramGrp>
    <pos>n</pos>
  </gramGrp>
  <def>one who is partly divine and partly human</def>
  <def>(in Gk myth, etc) the son of a god and a mortal woman,
       eg <mentioned>Hercules</><pron>"h3:kjUli:z</pron></def>
</entryFree>
```

The <entryFree> element also makes it possible to transcribe a dictionary using only phrase-level ("atomic") elements—that is, using no grouping elements at all. This can be desirable if the encoder wants a completely "flat" view, with no indication of or commitment to the association of one element with another. The following encoding uses no grouping elements, and keeps all rendition text:

" **biryani** or **biriani** (%bIrI"A:nI) any of a variety of Indian dishes...[from Urdu] [CED] "

```
<entryFree>
  <orth>biryani</orth> or
  <orth>biriani</orth>
  <pron>(%bIrI"A:nI)</pron>
  <def>any of a variety of Indian dishes...</def>
  <etym>[from <lang>Urdu</lang>]</etym>
</entryFree>
```

The declaration of <entryFree> is given above in section 12.2 ('The Structure of Dictionary Entries') on p. 274. It allows any elements defined for use within dictionary entries to appear within it, in any order.

```
<!-- 12.6:  Model class for unstructured dictionary entries   -->
<!-- This entity declares the class of elements defined       -->
<!-- specifically for use in dictionary entries, except       -->
<!-- those which are included in the phrase class. This       -->
<!-- class is used in defining the 'free' dictionary entry.   -->

<!ENTITY % x.dictionaryParts ''                                 >
<!ENTITY % m.dictionaryParts '%x.dictionaryParts case | colloc
           | def | eg | etym | form | gen | gramGrp | hom |
           hyph | itype | lbl | mood | number | orth | per |
           pos | pron | re | sense | stress | subc | superentry
           | syll | tns | tr | trans | usg | xr'               >
<!-- This fragment is used in sec. 12.1                       -->
```

Chapter 13

Terminological Databases

Terminological information generally resides in *terminology databases* (*TDBs*), but for SGML applications, these collections of data can be viewed as documents. A document containing terminological data is made up of *terminological entries*. Typically, a terminological entry treats a single concept and contains information on the assignment of single or multi-word terms to this concept. Bilingual and multilingual terminological entries deal with harmonized or very closely related concepts in two or more languages that are treated as functional equivalents in the context of a specific domain or subdomain. Terminological data can take the form of terminological databases (TDBs) or can be used to print hardcopy terminological documents, such as terminological dictionaries, technical vocabularies, or thesauri.

The TEI description of terminological data was originally designed primarily as a *terminology interchange format* (*TIF*) to allow users of terminology databases to exchange database records. In this guise it is called the *Electronic Terminology Interchange Format* (*E-TIF*). The exchange of database records is especially important in practice because the structure of terminological records varies considerably from TDB to TDB, reflecting differences of design and of user needs. Users of TDBs frequently need to interchange data in order to access expert information and to prevent the duplication of effort, but differences in software, hardware, and methodology complicate interchange. A universal interchange format is a crucial element in making interchange easier.

The tag set defined in this chapter may also be used to mark up documents for the purpose of printing *terminological dictionaries* and *vocabularies*, or exchanging them in electronic form. Printed terminological documents differ from terminological databases in that they are frequently divided into sections and subsections and include prose text in introductions, etc. When used for marking up printed documentation, we can speak of the tag set defined here as a *Print Terminology Interchange Format* (*P-TIF*).

Because printed terminological dictionaries differ from terminological databases, problems may arise if one attempts to use the same electronic document both for printing and to exchange records among databases. A printed terminological dictionary may contain material not suitably encoded for introduction into database records. Domain and subdomain information may be implied by the arrangement of <termEntry>s rather than by explicit domain specifications within the individual entries.

Other interchange difficulties include differences between term entry styles used in prescriptive and descriptive terminology work and problems arising from differences in the degree of detail used to classify data elements in different databases. (The term *data element* is used by terminologists to refer to 'the smallest defined individual items of information', regardless of whether they are represented as SGML elements, SGML attributes, or fields or columns in a database. That is the usage followed here.) Procedures for addressing these various problems are treated in more detail in another document, the *TEI / LISA / ISO - TIF — Terminology Interchange Format — A Tutorial* (1993). [1]

[1] This document is reprinted in *TermNet News*, no 40, 1993, pp 5-64; copies are also available from Infoterm, z.Hd. Herrn Dr. Gerhard Budin, Heinestraße 38, Postfach No. 130, A-1021 Vienna, Austria.

13.1 The Terminological Entry

The basic unit of terminology management is the *terminological entry*. A terminological entry documents information pertaining to a concept and generally speaking contains at least one *term*. In addition to the term, various kinds of descriptive and administrative data are recorded concerning the term, the concept to which it is assigned, and relationships to other terms and concepts. Administrative information supports the management of the terminology database or document.

A sample terminological entry consists of a series of entries like the following:

subject field appearance of materials
English term opacity
grammatical information, part of speech, English term noun
definition, English term degree of obstruction to the transmission of visible light
bibliographical source, English term information ASTM Standard E284
responsibility for English term information ASTM Technical Committee E12
German term Opazität
grammatical information, part of speech, German term noun
grammatical information, gender, German term feminine
definition, German term Maß für die Lichtdurchsichtigkeit
bibliographical source, German term information HFdn1983-382
responsibility, German term information DIN Technical Committee for paper products
French term opacité
grammatical information, part of speech, French term noun
grammatical information, gender, French term feminine
definition, French term rapport du flux lumineux incident au flux lumineux transmis ou réfléchi par un noircissement photographique
bibliographical source, French term information HJdi1986
responsibility, French term information C.I.R.A.D.

13.2 Tags for Terminological Data

The following sections define elements for use in tagging terminological data. The elements and attributes listed are based on empirical studies. The studies indicated the use of a wide variety of different data element types (data categories or database field types), but this variety can be reduced to a relatively small set of SGML elements and attributes expressing notions common to most, if not all, TDBs. Those elements and attributes are defined here. In addition, the global TEI attributes defined in section 3.5 ('Global Attributes') on p. 42, and the elements and attributes defined in chapter 6 ('Elements Available in All TEI Documents') on p. 119, can all be used in terminological applications.

When tagging terminological data, three elements constitute the set of *non-floating elements*: <term>, <otherForm>, and <descrip>. All other elements function as *floating elements*, including: <admin>, <note>, <gram>, <bibl>, <biblFull>, <date>, <table>, <formula>, <figure>, and the linking elements (<ptr>, <xptr>, <ref>, and <xref>). The rules for combining floating with non-floating elements are spelled out below in section 13.3.1 ('Nested Term Entries') on p. 316, and in section 13.3.2 ('Flat Term Entries Using Rules of Adjacency') on p. 316.

<term> contains a single-word, multi-word or symbolic designation which is regarded as a technical term. Attributes include:

type classifies the term using some typology.

<termEntry> contains a single complete entry for one concept expressed in one language and comprising one or more terms and their
associated descriptive and administrative data, or, in bilingual and multilingual terminology

work, two or more very closely related concepts comprising one or more terms in each language and their associated descriptive and administrative data. Attributes include:

type classifies the term entry using some typology, preferably the dictionary of data element types specified in ISO WD 12 620.

<tig> within a <termEntry> element, contains information elements associated with a single term. Attributes include:

type classifies the <tig> using some typology, preferably the dictionary of data element types specified in ISO WD 12 620.

<otherForm> contains an alternate designation for the concept treated by the term entry, such as a synonym. Attributes include:

type classifies the <otherForm> using some typology, preferably the dictionary of data element types specified in ISO WD 12 620.

<ofig> within a <tig> element, contains information elements relating to a single <otherForm>. Attributes include:

type classifies the other-form information group according to some convenient typology, preferably the dictionary of data element types specified in ISO WD 12 620.

<gram> within an entry in a dictionary or a terminological data file, contains grammatical information relating to a term, word, or form. Attributes include:

type classifies the grammatical information given according to some convenient typology — in the case of terminological information, preferably the dictionary of data element types specified in ISO WD 12 620. Suggested values include:

pos part of speech (any of the word classes to which a word may be assigned in a given language, based on form, meaning, or a combination of features, e.g. noun, verb, adjective, etc.)

gen gender (formal classification by which nouns and pronouns, and often accompanying modifiers, are grouped and inflected, or changed in form, so as to control certain syntactic relationships)

num number (e.g. singular, plural, dual, ...)

animate animate or inanimate

proper proper noun or common noun

<descrip> within a <termEntry> element, contains a definition, context or explanation used to explain or define the concept represented by a <term> or an <otherForm>. Attributes include:

type classifies the description using some convenient typology, preferably the dictionary of data element types specified in ISO WD 12 620. Suggested values include:

definition The description provides all the information needed to differentiate one concept from all other related concepts in the given domain.

<admin> within a <termEntry> element, contains administrative information pertaining to data management and documentation of the entry. Attributes include:

type identifies the administrative event or information using some typology, preferably the dictionary of data element types specified in ISO WD 12 620. Suggested values include:

responsibility The <admin> element identifies the agency or individual responsible for the data element or entry.

created The <admin> element describes the creation of the data element or entry.

updated The <admin> element describes the update or modification of the data element or entry.

approved The <admin> element describes the final approval of the data element or entry.

domain The element indicates the subject area to which a concept pertains.

subdomain The element indicates the subdomain of the subject area to which the concept pertains.

As indicated, these elements all possess a type attribute, used to classify the generic elements so as to match the classifications used by TDBs. The type attributes allow specific items of information not defined in the DTD to be tagged as one of the defined elements with an appropriate type value. The possible values of type thus constitute a sizable open list.

At the time of publication, work is under way in ISO Technical Committee 37, Sub-Committee 3, Working Group 1 to compile an official dictionary of data element types (data categories) for use in terminology work, which will eventually provide the core for a complete list of type attribute values. This data element dictionary will appear as ISO 12 620. The attribute values that occur in the examples shown in this chapter represent a subset of those that will be defined in ISO 12 620.

The <ofig> and <otherForm> elements are not necessary if each potential <otherForm> element is recast as a term in its own <tig>. For example, a term could be placed in a <tig type=synonym>.

When the base tag set described in this chapter is used, the following attributes are added to the set of global attributes:

group indicates the group (term and related elements) to which this element should be associated by specifying a string matching the n attribute value on an appropriate element.

depend indicates the parent element to which this element should be associated by specifying a string matching the n attribute value on an appropriate element.

grpPtr indicates the group (term and related elements) to which this element should be associated by specifying its unique identifier, where this is available.

depPtr indicates the parent element to which this element should be associated by specifying its unique identifier, where this is available.

For discussion of the usage of these attributes, see below, section 13.3.2 ('Flat Term Entries Using Rules of Adjacency') on p. 316.

Among the TEI core elements, the following are most likely to be found necessary in encoding terminological data; for fuller descriptions see the appropriate sections in chapter 6 ('Elements Available in All TEI Documents') on p. 119. In the case of the <date> element, it should be noted that the ISO format (YYYY-MM-DD) is preferred for terminology entries.

<note> contains a note or annotation.

<ref> defines a reference to another location in the current document, in terms of one or more identifiable elements, possibly modified by additional text or comment.

<ptr> defines a pointer to another location in the current document in terms of one or more identifiable elements.

<xref> defines a reference to another location in the current document, or an external document, using an extended pointer notation, possibly modified by additional text or comment.

<xptr> defines a pointer to another location in the current document or an external document.

<date> contains a date in any format.

<bibl> contains a loosely-structured bibliographic citation of which the sub-components may or may not be explicitly tagged.

<biblStruct> contains a structured bibliographic citation, in which only bibliographic subelements appear and in a specified order.

<biblFull> contains a fully-structured bibliographic citation, in which all components of the TEI file description are present.

<table> contains text displayed in tabular form, in rows and columns.

<figure> indicates the location of a graphic, illustration, or figure.

<formula> contains a mathematical or other formula.

Like all other elements defined in the TEI DTDs, all elements in the base tag set for terminology possess the following global attributes:

lang indicates the language of the element content, usually using a two- or three-letter code from ISO 639.

n gives a number (or other label) for an element, which is not necessarily unique within the document.

id provides a unique identifier for the element bearing the ID value.

Using the tags defined here, the example given above in section 13.1 ('The Terminological Entry') on p. 312 might be tagged thus:[2]

```
<!-- Example 2a:  Nested Term Entry -->
<termEntry>
     <admin type='domain'> appearance of materials </admin>
     <tig lang=en>
          <term> opacity </term>
          <gram type=pos> n </gram>
          <descrip type='definition'> degree of obstruction to the
          transmission of visible light </descrip>
          <ptr type='bibliographic' target='ASTM.E284'>
          <admin type='responsibility' resp='ASTM E12'> </admin>
     </tig>
     <tig lang=de>
          <term> Opazit&auml;t </term>
          <gram type=pos> n </gram>
          <gram type=gen> f </gram>
          <descrip type='definition'> Ma&szlig; f&uuml;r die
          Lichtdurchsichtigkeit </descrip>
          <ref type='bibliographic' target='HFdn1983'> p. 383 </ref>
          <admin type='responsibility' resp='DIN TC for paper
          products'></admin>
     </tig>
     <tig lang=fr>
          <term> opacit&eacute; </term>
          <gram type=pos> n </gram>
          <gram type=gen> f </gram>
          <descrip type='definition'> rapport du flux lumineux
          incident au flux lumineux transmis ou r&eacute;fl&eacute;chi
          par un noircissement photographique </descrip>
          <ptr type='bibliographic' target='HJdi1986'>
          <admin type='responsibility' resp='C.I.R.A.D.'> </admin>
     </tig>
</termEntry>
```

Both the <ptr type='bibliographic' target='ASTM.E284'> and <ref type='bibliographic' target='HFdn1983'> elements in the example indicate links to complete bibliographical entries included in the back matter element of the same document. 'HFdn1983' is a source reference code for a book, generated according to ISO/TC 37 WI 18, *Coding of Bibliographic References in Terminology Work and Terminography* (1991). Its full bibliographic record would be:

```
<!-- Example 2b:  Full Bibliographic Entry -->
<biblFull>
       <titleStmt id=HFdn1983>
              <editor> Henry G. Freeman </editor>
              <title> W&ouml;rterbuch technischer Begriffe mit 4300
              Definitionen nach DIN </title>
       </titleStmt>
       <editionStmt>
              <edition> III </edition>
       </editionStmt>
       <extent> 703 pp </extent>
       <publicationStmt>
              <creation.date> 1983 <creation.date>
              <publisher> Beuth Verlag GmbH </publisher>
              <place> Berlin and K&ouml;ln </place>
              <date> 1983 </date>
       </publicationStmt>
       <sourceDesc><p>Compiled for the standards of the DIN (Deutsches
```

[2]In this example, as in the others, white space has been liberally used for the sake of legibility; in practice most actual encodings would use less white space.

```
                Institut f&uuml;r Normung).</p>
        </sourceDesc>
</biblFull>
```

Further examples, including alternate encodings of this term entry, are given below in section 13.3.2 ('Flat Term Entries Using Rules of Adjacency') on p. 316, and section 13.3.3 ('Flat Term Entries Using Group and Depend Attributes') on p. 317.

The formal definition of these elements depends on which style of markup is being used; for discussion of the two styles, see the following section, 13.3 ('Basic Structure of the Terminological Entry') on p. 316. For the formal declarations for the two styles, see sections 13.4.1 ('DTD Fragment for Nested Style') on p. 321, and 13.4.2 ('DTD Fragment for Flat Style') on p. 322.

13.3 Basic Structure of the Terminological Entry

A terminological entry is identified with the <termEntry> tag and contains one or more terms marked with the tag <term>, which may appear with associated SGML elements. A single term and its associated SGML elements (such as <gram>, <descrip>, <admin>) constitute a *term information group*, <tig>. A <termEntry> may be made up of one or more <tig>s.

There are two structural descriptions for <termEntry>s:

- *nested* <termEntry>s
- *flat* <termEntry>s

The nested structure is preferred, especially for interchange with unknown partners. The flat structure provides an option that can be used between interchange partners whose systems exhibit fairly similar structures. The flat structure may also be used as an intermediate form for systems making the transition to the nested format.

13.3.1 Nested Term Entries

A nested <termEntry> uses SGML to represent the hierarchical relationships implicit in the terminological entry by utilizing the following principles of embedding and adjacency.

- Rule of embedding in nested term entries: Elements that constitute a part of another element are embedded inside the parent element.
- Rules of adjacency in nested term entries:

  N1 Any element that appears in a <termEntry> outside a <tig> applies to the entire <termEntry>.

  N2 Any element that appears in a <tig> before the <term> element applies to the entire <tig>.

  N3 Any floating element that appears after a non-floating element (i.e., after <term>, <otherForm> or <descrip>) and before the next non-floating element, refers to the immediately preceding non-floating element unless otherwise indicated using the depend attribute. (See section 13.3.3 ('Flat Term Entries Using Group and Depend Attributes') on p. 317, for a full discussion of the depend attribute.)

The conversion routine that creates the nested entry infers the language of the <tig> from the language of the <term>, a process that can be construed as "upward inheritance" from <term> to <tig>. Standard TEI "downward inheritance" applies for all the elements embedded in the <tig>: their language is that of the <tig>, unless this default value is overridden by stating a new value.

An example of a nested term entry was given in section 13.2 ('Tags for Terminological Data') on p. 312.

13.3.2 Flat Term Entries Using Rules of Adjacency

The flat terminological entry does not use the <tig> element to enclose a term and its associated elements. Instead, it provides other mechanisms to express the relationships that occur within and among entries in a TDB, while at the same time allowing the different types of entries found

in different source TDBs to be represented in very natural ways. The difference between the nested and flat terminological entries is that, while both can express the same information, the nested structure represents the logical hierarchy implicit within the entry by embedding elements in one another, while the flat entry does not represent the logical hierarchy within the entry in this way. Since many existing TDBs do not overtly indicate any hierarchical structure such as that represented in a nested entry, the flat entry may be more apt to reflect the organization of data elements within an entry found in the particular source TDB, whereas the nested entry more obviously characterizes an ideal abstract structure of the term entry. In flat entries, terms and their associated elements are grouped by means of the following rules of adjacency:

Rules of adjacency in flat <termEntry>s:

F1 Any element that appears in a <termEntry> before the first <term> is assumed to apply to the entire <termEntry>.

F2 Any floating element that appears after a non-floating element (i.e., after <term>, <otherForm> or <descrip>) and before the next non-floating element refers to the immediately preceding non-floating element unless otherwise indicated using the depend attribute. (See section 13.3.3 ('Flat Term Entries Using Group and Depend Attributes') on p. 317, for a full discussion of the depend attribute.)

Encoded using the flat style, the example given in section 13.2 ('Tags for Terminological Data') on p. 312, might look like this:

```
<!-- Example 3:  Flat <TermEntry>                                  -->
<termEntry>
     <admin type='domain'> appearance of materials </admin>
     <term lang=en> opacity </term>
     <gram type=pos> n </gram>
     <descrip type='definition'> degree of obstruction to the
     transmission of visible light </descrip>
     <ptr type='bibliographic' target='ASTM.E284'>
     <admin type='responsibility' resp='ASTM E12'></admin>
     <term lang=de> Opazit&auml;t </term>
     <gram type=pos> n </gram>
     <gram type=gen> f </gram>
     <descrip type='definition'> Ma&szlig; f&uuml;r die
      Lichtdurchsichtigkeit
     </descrip>
     <ref type='bibliographic' target='HFdn1983'> p. 383 </ref>
     <admin type='responsibility' resp='DIN TC for paper products'>
     </admin>
     <term lang=fr> opacit&eacute; </term>
     <gram type=pos> n </gram>
     <gram type=gen> f </gram>
     <descrip type='definition'> rapport du flux lumineux
     incident au flux lumineux transmis ou r&eacute;fl&eacute;chi
     par un noircissement photographique </descrip>
     <ptr type='bibliographic' target='HJdi1986'>
     <admin type='responsibility' resp='C.I.R.A.D.'> </admin>
</termEntry>
```

13.3.3 Flat Term Entries Using Group and Depend Attributes

In practice, there are term entries where elements are ordered in such a way that the rules of adjacency cannot be used. For instance, in Example 3 the <ptr> and <ref> linking elements refer to the immediately preceding <descrip> information. The <admin type='responsibility'> elements as represented here also refer to the <descrip> element. It may, however, be desirable for the bibliographic reference to refer not only to the quoted material in the descriptive element, but also to the term itself. Because the second rule of adjacency dictates that all floating elements following a non- floating element refer to that non-floating element, a mechanism is required to "point" to the <term> if the floating element depends on the <term> itself.

There are also other exceptions to the adjacency rules: in some term entries elements are associated with a <term> other than the immediately preceding <term>. Such entries may be called *discontiguous flat term entries*, since the constituents of a term information group may not be adjacent. In such entries, information pertaining to the entire terminological entry may not always appear at the beginning of the entry (i.e., prior to the introduction of a term).

Such an entry might be encoded as follows:

```
<!-- Example 4:  Discontiguous Flat <termEntry>  -->
<termEntry n=texyz>
     <term lang=en n=1> opacity </term>
     <gram type=pos depend=1> n </gram>
     <term lang=de n=2> Opazit&auml;t </term>
     <gram type=pos depend=2> n </gram>
     <gram type=gen depend=2> f </gram>
     <term lang=fr n=3> opacit&eacute; </term>
     <gram type=pos depend=3> n </gram>
     <gram type=gen depend=3> f </gram>

     <descrip type='definition' group=1 n=endes1> degree of
        obstruction to the transmission of visible light  </descrip>
     <descrip type='definition' group=2 n=dedes1> Ma&szlig; f&uuml;r die
        Lichtdurchsichtigkeit  </descrip>
     <descrip type='definition' group=3 n=frdes1> rapport du
        flux lumineux incident au flux lumineux transmis ou
        r&eacute;fl&eacute;chi par un noircissement photographique
        </descrip>
     <ptr type='bibliographic' depend=endes1 target='ASTM.E284'>
     <admin type='responsibility' depend=endes1 resp='ASTM E12'> </admin>
     <ref type='bibliographic' depend=dedes1 target='HFdn1983'>
        p. 383 </ref>
     <admin type='responsibility' depend=dedes1
        resp='DIN.TC.for.paper'></admin>
     <ptr depend=frdes1 type='bibliographic' target='HJdi1986'>
     <admin type='responsibility' depend=frdes1
        resp='C.I.R.A.D.'> </admin>
     <admin type='domain' depend=texyz> appearance of materials
     </admin>
</termEntry>
```

In the above example, depend elements indicate that the material tagged with this attribute is related to the targeted element. The group elements indicate that the information so marked is part of an implicit <tig>, i.e. that it pertains either to the term or to the entire implicit <tig>. Items linked to other elements by depend do not require the group attribute because they are associated with the group already by virtue of their relation to elements that are themselves associated with the group.

So as to describe appropriate relationships in discontiguous flat <termEntry>s, it is necessary to define a pointing mechanism that allows any non-adjacent element to be related to an implicit term information group and therefore to the <term> with which it is associated or to some other specific element.

Two methods are provided to represent this association. For terminology files in which unique identifiers for all <term> elements cannot be assumed (as will often be the case in interchange), the group and depend attributes should be used. For terminology files in which unique SGML identifiers can be provided, the grpPtr and depPtr attributes should be used. The two pairs of attributes have identical significance as far as the association of elements is concerned.

The group attribute associates an element with a specific term, or with an implicit term information group: its value must be the same as the n attribute on the <term> element being pointed to. During interchange, the group attribute would be used to extract and assemble all the elements related to a specific term information group from a discontiguous flat <termEntry> by matching them to the n attributes on the terms. The group pointer accounts for the kind of relationship represented by the principle of embeddedness within a <tig> in a nested term entry.

The depend attribute associates an element with some other specific element: its value must be the same as the n attribute on the element being pointed to. As shown in the last line of Example 4, the depend attribute can also point to the entire terminological entry by targeting a value of n indicated in the <termEntry> element. If for any reason the grammatical information pertaining to a term does not follow the term immediately, this information must be linked to the term with the depend attribute.

In terms of the extended pointer notation defined in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331, the specification group=2 is synonymous with HERE ANCESTOR (1 TERMENTRY) DESCENDANT (1 TERM N 2), and the specification depend=3 is synonymous with HERE ANCESTOR (1 TERMENTRY) DESCENDANT (1 * N 3).

To summarize the behavior of group and depend, the group attribute identifies an implicit <tig>, whereas the depend attribute implies relatedness. If there is any ambiguity with respect to the rules of adjacency, one should use depend.

In Example 4, the English term 'opacity' is identified as n=1, and all other elements associated with this <tig> are marked as group=1; in German, the term and all its associated elements are identified as n=2 and group=2, respectively; in French, the term and associated elements are marked group=3. Since the bibliographical references are displaced from the descriptive information with which they are associated, the descriptions are identified with n=endes1, n=dedes1, and n=frdes1, respectively. The <ptr> and <ref> elements are then identified with depend attributes that target the appropriate descriptions. Even if the elements in the entry were adjacent to each other in the entry, this convention would be essential if one wanted to indicate that the source applied to the <term> and hence to the entire <tig>, rather than just to the <descrip> element itself.

13.3.4 References between Term Entries

Terminology documents utilize a variety of cross-references between <termEntry>s, for instance to link to bibliographic entries or between equivalents in different languages, synonyms and related terms and concepts. These references are usually implemented using the TEI linking elements <ptr> and <ref>, together with a value of the attribute type. If, as is the case with the reference to ASTM E284, the total bibliographic source description is contained in the "target" element of the linking element, use <ptr>. If, on the other hand, a page number is included, this page number must appear as the content of a linking element introduced by the <ref> element.

Examples:

```
<ptr type='bibliographic' target='ASTM.E284'>
```

or

```
<ref type='bibliographic' target='HFdn1983'> p. 383 </ref>
```

If the full bibliographical citation is included in the <termEntry> itself, linking elements are unnecessary and the citation can be marked using the <bibl>, <biblStruct>, or <biblFull> elements. For further discussion of bibliographic citations and references, see section 6.10 ('Bibliographic Citations and References') on p. 162.

13.4 Overall Structure of Terminological Documents

To enable the base tag set for terminology, a parameter entity *TEI.terminology* must be declared within the document type subset, the value of which is INCLUDE, as further described in section 3.3 ('Invocation of the TEI DTD') on p. 39. A document using this base tag set and no other additional tag sets will thus begin as follows:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
     <!ENTITY % TEI.terminology 'INCLUDE' >
]>
```

This declaration makes available all of the elements described in this chapter, in addition to the core elements described in chapter 6 ('Elements Available in All TEI Documents') on

p. 119. The default structure for terminological documents is similar to that defined by chapter 7 ('Default Text Structure') on p. 183: within the <TEI.2> element they contain a <teiHeader> and a <text>. The <text> element, in turn, contains as usual a <body> element, optionally preceded by a <front> and followed by a <back>. The <body> may contain a series of <termEntry> elements, which may optionally be grouped into sections tagged with the same elements (, <div0>, <div1>, etc.) as defined in section 7.1 ('Divisions of the Body') on p. 185.

<text> contains a single text of any kind, whether unitary or composite, for example a poem or drama, a collection of essays, a novel, a dictionary, or a corpus sample.
<body> contains the whole body of a single unitary text, excluding any front or back matter.
** ** contains a subdivision of the front, body, or back of a text.
<div0> contains the largest possible subdivision of the body of a text.
<div1> contains a first-level subdivision of the front, body, or back of a text (the largest, if <div0> is not used, the second largest if it is).
<div2> contains a second-level subdivision of the front, body, or back of a text.
<div3> contains a third-level subdivision of the front, body, or back of a text.
<div4> contains a fourth-level subdivision of the front, body, or back of a text.
<div5> contains a fifth-level subdivision of the front, body, or back of a text.
<div6> contains a sixth-level subdivision of the front, body, or back of a text.
<div7> contains the smallest possible subdivision of the front, body or back of a text, larger than a paragraph.

In order to support both the flat and the nested styles of markup, three distinct DTD fragments for terminology are provided.

- teiterm2
- teite2n
- teite2f

In file *teiterm2.dtd*, the top-level elements for the terminology base are defined, and a subordinate parameter entity, *termtags* is defined and referred to. By default, this entity refers to file *teite2n.dtd*, which defines the DTD for nested markup; if the flat style of markup is to be used, the document's DTD subset should define *termtags* as referring to the file *teite2f.dtd*, as shown in the examples in section 13.3.2 ('Flat Term Entries Using Rules of Adjacency') on p. 316.

```
<!-- 13.4: TEIterm2.DTD: Base tag set for terminological data -->
<!-- Text Encoding Initiative: Guidelines for Electronic      -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.    -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy    -->
<!-- in any form is granted, provided this notice is          -->
<!-- included in all copies.                                  -->

<!-- These materials may not be altered; modifications to     -->
<!-- these DTDs should be performed as specified in the       -->
<!-- Guidelines in chapter "Modifying the TEI DTD."           -->

<!-- These materials subject to revision. Current versions    -->
<!-- are available from the Text Encoding Initiative.         -->
<!-- First, embed the default text structure elements.        -->

<![ %TEI.singleBase [
<!ENTITY % TEI.structure.dtd system 'teistr2.dtd'              >
%TEI.structure.dtd;
]]&nil;>


<!ENTITY % termtags system 'teite2n.dtd'                       >
%termtags;
```

In file *teiterm2.ent*, terminology-specific extensions to the TEI element class system are defined, including the classes *terminology*, *comp.terminology*, *terminologyInclusions*, and *terminologyMisc*.

```
<!-- 13.4: TEIterm2.ent: Base tag set for terminological data -->
<!-- Text Encoding Initiative: Guidelines for Electronic      -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.    -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy    -->
<!-- in any form is granted, provided this notice is          -->
<!-- included in all copies.                                  -->

<!-- These materials may not be altered; modifications to     -->
<!-- these DTDs should be performed as specified in the       -->
<!-- Guidelines in chapter "Modifying the TEI DTD."           -->

<!-- These materials subject to revision. Current versions    -->
<!-- are available from the Text Encoding Initiative.         -->


<!ENTITY % x.comp.terminology ''                                  >
<!ENTITY % m.comp.terminology '%x.comp.terminology termEntry'     >
<!ENTITY % seq '(%m.common; | %m.comp.terminology;)* '            >
<!ENTITY % mix.terminology '| %m.comp.terminology'                >

<!ENTITY % x.terminologyInclusions ''                             >
<!ENTITY % m.terminologyInclusions '%x.terminologyInclusions
           date | dateStruct | note | ptr | ref | xptr | xref'  >
<!ENTITY % x.terminologyMisc ''                                   >
<!ENTITY % m.terminologyMisc '%x.terminologyMisc admin |
           descrip'                                               >

<!-- Add attributes to the set of global attributes:          -->

<!ENTITY % a.terminology '
          group              CDATA              #IMPLIED
          grpPtr             IDREF              #IMPLIED
          depend             CDATA              #IMPLIED
          depPtr             IDREF              #IMPLIED'      >
```

13.4.1 DTD Fragment for Nested Style

In file *teite2n.dtd* the following definitions are found, which define the elements used in the nested markup style:

```
<!-- 13.4.1:  Elements for nested-style terminological data   -->

<!-- The nested structure is used for data interchange and    -->
<!-- represents a canonical structured form for terminology   -->
<!-- entries, which differs from the less structured forms    -->
<!-- frequently used to store data in terminological          -->
<!-- databases.                                               -->

<!ELEMENT termEntry     - O  ((%m.terminologyMisc)*, tig+)

                                              +(%m.terminologyInclusions)
                                                              >
<!ATTLIST termEntry          %a.global;
          type               CDATA              #IMPLIED       >
<!-- Notes, descrip(s) and admin(s) are allowed in the        -->
<!-- termEntry to provide documentation that applies to the   -->
<!-- whole entry.                                             -->
```

```
<!-- tig='term information group'                                -->

<!-- ofig='otherform information group'                          -->


<!ELEMENT tig           - O  ((%m.terminologyMisc)*, (term,
                             gram*), (%m.terminologyMisc)*,
                             ofig*)                               >
<!ATTLIST tig                %a.global;
          type               CDATA              #IMPLIED        >
<!-- Order is significant: term, descrip(s), ofig(s) or          -->
<!-- otherform(s)                                                -->


<!ELEMENT ofig          - O  ((%m.terminologyMisc)*,
                             (otherForm, gram*),
                             (%m.terminologyMisc)*)               >
<!ATTLIST ofig               %a.global;
          type               CDATA              #IMPLIED        >
<!ELEMENT otherForm     - O  (%paraContent;)                      >
<!ATTLIST otherForm          %a.global;
          type               CDATA              #IMPLIED        >
<!ELEMENT descrip       - O  (%paraContent;)                      >
<!ATTLIST descrip            %a.global;
          type               CDATA              #IMPLIED        >
<!ELEMENT admin         - O  (%paraContent;)                      >
<!ATTLIST admin              %a.global;
          type               CDATA              #IMPLIED
          date               CDATA              %ISO-date
          resp               CDATA              #IMPLIED        >
<!-- We define a.dictionaries as the empty string, since we     -->
<!-- are not now using the tag set for dictionaries.            -->

<!ENTITY % a.dictionaries ''                                      >
<!ELEMENT gram          - O  (%paraContent;)                      >
<!ATTLIST gram               %a.global;
                             %a.dictionaries;
          type               CDATA              #IMPLIED        >
```

13.4.2 DTD Fragment for Flat Style

In file *teite2f.dtd* the following definitions, which provide support for the flat markup style, are found:

```
<!-- 13.4.2:  Elements for flat-style terminological data       -->
<!-- The flat structure is used to represent a variety of       -->
<!-- terminology documents that occur in practice and which     -->
<!-- do not follow the form of the nested interchange           -->
<!-- format. The flat representation allows for a less rigid    -->
<!-- structure, but provides a rich mechanism for reflecting    -->
<!-- inter-element relations.                                   -->


<!-- The declaration of termEntry enforces appearance of at     -->
<!-- least one term element in a termEntry, which may be        -->
<!-- preceded by descrip, admin, note, otherform, or gram.      -->
<!-- There may be multiple notes, admins, descrips              -->
<!-- otherforms, and grams appearing in any order. xRef,        -->
<!-- date, biblRef can appear in all positions in termEntry.    -->
```

```
<!ELEMENT termEntry     - O  ( (%m.terminologyMisc | otherform
                             | gram |
                             %m.terminologyInclusions)*, (term,
                             (%m.terminologyMisc | otherform |
                             gram | %m.terminologyInclusions)*
                             )+ )                                >
<!ATTLIST termEntry          %a.global;
          type               CDATA               #IMPLIED       >
<!ELEMENT otherForm     - O  (%paraContent;)                     >
<!ATTLIST otherForm          %a.global;
          type               CDATA               #IMPLIED       >
<!ELEMENT descrip       - O  (%paraContent;)                     >
<!ATTLIST descrip            %a.global;
          type               CDATA               #IMPLIED       >
<!ELEMENT admin         - O  (%paraContent;)                     >
<!ATTLIST admin              %a.global;
          type               CDATA               #IMPLIED
          date               CDATA               %ISO-date
          resp               CDATA               #IMPLIED       >
<!-- We define a.dictionaries as the empty string, since we   -->
<!-- are not now using the tag set for dictionaries.          -->

<!ENTITY % a.dictionaries ''                                     >
<!ELEMENT gram          - O  (%paraContent;)                     >
<!ATTLIST gram               %a.global;
                             %a.dictionaries;
          type               CDATA               #IMPLIED       >
```

13.5 Additional Examples of Term Entries

The tag set defined in this chapter is designed to accommodate the variety of structures that occur in TDBs; this section shows the how the same information may be encoded in different ways, depending on local convenience or preferences. Example 5 gives an entry from an ISO terminological standard. Example 6 treats this English-French equivalent pair as a single nested terminological entry, whereas Example 7 splits the information into two nested entries with cross-references. Example 8 shows the same data as a flat terminological entry with adjacent elements, whereas Example 9 groups the elements according to element type, which requires the use of pointers in order to reconstruct the implicit terminological information group from discontiguous elements.

The interchange of terminological data between TDBs requires an export routine (to E-TIF) and an import routine (from E-TIF). For interchange between unknown partners, it may be desirable to normalize the encoding method rather than allow all the options presented in this section. The effect of normalization would be that import routines become easier to implement while export routines become more difficult to implement. At the time of this publication, work is under way in ISO Technical Committee 37, Subcommittee 3, Working Group 3 on a normalized version of E-TIF called ISO [DIS] 12 200. Some aspects of normalization under consideration are to use only the nested representation and avoid the use of the following options: divisions within the <body>, the <otherForm> element, the group and depend attributes, elements before the <term> element in a <tig>, inclusion exceptions other than <ptr> and <xptr>, and paragraph content other than #PCDATA in the elements <admin> and <gram>.

13.5.1 Example Term Entry from ISO 472

The following term entry is taken from ISO 472:1988, *Plastics — Vocabulary*, Bilingual edition (Geneva: ISO, 1988), p. 84. The original uses typographic characteristics to represent different data element types within the term entry, not all of which have been retained in the reproduction of this sample. As prescribed by ISO layout guidelines,[3] the original text is printed in Helvetica, with English and French information presented in two parallel columns; head terms appear in bold face, notes in a smaller font size than the main text, and terms referred to in the cross references are printed in italics.

thermal degradation The entirety of all deleterious chemical modifications of plastic at elevated temperature. NOTE — It is essential to report the temperature and other environmental conditions at which the phenomenon is studied.
See also *ageing*, *degradation* and *deterioriation*.

décomposition thermique Ensemble de toutes les modifications chimiques nuisibles d'un plastique à température élevée. NOTE — Il est essentiel d'indiquer la température et les autres conditions d'environnement dans lesquelles le phénomène est étudié.
Voir aussi *viellissement*, *dégradation* et *détérioration*.

13.5.2 The Example Treated as a Single Term Entry in Nested Form

This treatment assumes that both the English and French terms are treated together in the same entry. The elements grouped together at the top of the term entry apply to the entire entry. Only the first of the three cross-referenced terms is included in this example; it is represented by a <ptr> link which targets a term entry (related concept) contained in the same document. The id values used here are purely arbitrary.

```
<termEntry id=te84.11>
    <admin type='domain'> plastics </admin>
    <ref type='bibliographic' target='ISO.472-1988'> p. 84 </ref>
    <admin type='creation' date='1988'
           resp='ISO/TC 61, Plastics'> </admin>
    <ptr type='relatedTerm' target='te04.06'>

    <tig lang=en>
          <term> thermal degradation </term>
          <gram type=pos> n </gram>
          <descrip type='definition'> The entirety of all
          deleterious chemical modifications of plastic at
          elevated temperature. </descrip>
          <note> It is essential to report the temperature and
          other environmental conditions at which the phenomenon
          is studied.  </note>
     </tig>

     <tig lang=fr>
          <term> d&eacute;composition thermique </term>
          <gram type=pos> n </gram>
          <gram type=gen> f </gram>
          <descrip type='definition'> Ensemble de toutes les
          modifications chimiques nuisibles d'un plastique &agrave;
          temp&eacute;rature &eacute;lev&eacute;e. </descrip>
          <note> Il est essentiel d'indiquer la temp&eacute;rature et
          les autres conditions d'environnement dans lesquelles le
          ph&eacute;nom&egrave;ne est &eacute;tudi&eacute;. </note>
     </tig>
<termEntry>
```

[3] ISO 10241, *Preparation and layout of international terminology standards*, 1993.

```
<!-- Referenced term entry: -->

<termEntry id=te04.06>
     <tig lang=en>
          <term> ageing </term>
          ...
     </tig>

     <tig lang=fr>
          <term> vieillissement </term>
          ...
     </tig>
 </termEntry>
```

13.5.3 The Example Treated as Two Separate Term Entries in Nested Form

This example takes cognizance of the fact that some TDBs treat each term in a single <termEntry> instead of grouping all the information for a single concept into a single <termEntry>. The rationale behind this approach is frequently that no two languages truly provide harmonized concepts, although in the case of standardized terminology it can generally be assumed that concepts have been harmonized. The significant difference in encoding that occurs in this type of system is that <ptr> linking elements are required more frequently to link to term equivalents and related terms in other entries in the same document. Since there is only one <tig> in each entry, the <ptr> element could come at the beginning, as shown in the previous example, or inside the <tig> as shown below.

```
<termEntry id=te84.11.en>

     <admin type='domain'> plastics </admin>
     <ref type='bibliographic' target='ISO.472-1988'> p. 84 </ref>
     <admin type='creation' date='1988' resp='ISO/TC 61, Plastics></admin>

     <tig lang=en>
          <term> thermal degradation </term>
          <gram type=pos> n </gram>
          <descrip type='definition'> The entirety of all
          deleterious chemical modifications of plastic at
          elevated temperature. </descrip>
          <note> It is essential to report the temperature and
          other environmental conditions at which the phenomenon
          is studied.  </note>
          <ptr type='relatedTerm' target='te04.06.en'>
          <ptr type='equivalent' lang=fr target='te84.11.fr'>
     </tig>

</termEntry>

<termEntry id=te84.11.fr>

      <admin type='domain'> plastics </admin>
      <ref type='bibliographic' target='ISO 472-1988'> p. 84 <ref>
      <admin type='creation' date='1988'
             resp='ISO/TC 61, Plastics'> </admin>

     <tig lang=fr>
          <term> d&eacute;composition thermique </term>
          <gram type=pos> n </gram>
          <gram type=gen> f </gram>
          <descrip type='definition'> Ensemble de toutes les
```

```
          modifications chimiques nuisibles d'un plastique &agrave;
          temp&eacute;rature &eacute;lev&eacute;e. </descrip>
          <note> Il est essentiel d'indiquer la temp&eacute;rature et
          les autres conditions d'environnement dans lesquelles le
          ph&eacute;nom&grave;ne est &eacute;tudi&eacute;. </note>
          <ptr type='relatedTerm' target='te04.06.fr'>
          <ptr type='equivalent' lang=en target='te84.11.en'>

     </tig>

<termEntry>

<!-- Referenced term entry: -->

<termEntry id=te04.06.en>
     <tig lang=en>
          <term> ageing </term>
          ...
     </tig>
</termEntry>

<termEntry id=te04.06.fr>
     <tig lang=fr>
          <term> vieillissement </term>
          ...
     </tig>
</termEntry>
```

13.5.4 The Example Treated as a Flat Term Entry Using Adjacency Rules

This version of Example 5 uses a flat style of encoding, following the pattern of many existing TDBs; elements associated with a given term follow it immediately:

```
<termEntry id=te84.11>
     <admin type='domain'> plastics </admin>
     <ref type='bibliographic' target='ISO.472-1988'> p. 84 </ref>
     <admin type='creation' date='1988'
            resp='ISO/TC 61, Plastics'> </admin>
     <term lang=en> thermal degradation </term>
     <gram type=pos> n </gram>
     <descrip type='definition'> The entirety of all deleterious
     chemical modifications of plastic at elevated temperature.
     </descrip>
     <note> It is essential to report the temperature and other
     environmental conditions at which the phenomenon is studied.
     </note>
     <term lang=fr> d&eacute;composition thermique </term>
     <gram type=pos> n </gram>
     <gram type=gen> f </gram>
     <descrip type='definition'> Ensemble de toutes les
     modifications chimiques nuisibles d'un plastique &agrave;
     temp&eacute;rature &eacute;lev&eacute;e. </descrip>
     <note> Il est essentiel d'indiquer la temp&eacute;rature et les
     autres conditions d'environnement dans lesquelles le
     ph&eacute;nom&egrave;ne est &eacute;tudi&eacute;. </note>
     <ptr type='relatedTerm' target='te04.06'>
<termEntry>

<!-- Referenced term entry: -->
```

```
<termEntry id=te04.06>
     <term lang=en> ageing </term>
     ...
     <term lang=fr> vieillissement </term>
     ...
</termEntry>
```

13.5.5 The Example Treated as a Flat Term Entry Not Using Adjacency Rules

Many translation-oriented terminologists who work with half-screen popup windows prefer the following layout because it enables them to see the various <term> options at the top part of their display window without having to scroll into the body of the <termEntry>. Note in this case that the <ref> element links the bibliographic information to the entire entry.

```
<termEntry id=te84.11 n=te84.11>
     <term lang=en n=1> thermal degradation </term>
     <gram type=pos depend=1> n </gram>
     <term lang=fr n=2> d&eacute;composition thermique
     </term>
     <gram type=pos depend=2> n </gram>
     <gram type=gen depend=2> f </gram>

     <descrip type='definition' group=1> The entirety of all
     deleterious chemical modifications of plastic at elevated
     temperature. </descrip>
     <descrip type='definition' group=2> Ensemble de toutes les
     modifications chimiques nuisibles d'un plastique &agrave;
     temp&eacute;rature &eacute;lev&eacute;e. </descrip>
     <note group=1> It is essential to report the temperature and
     other environmental conditions at which the phenomenon is
     studied. </note>
     <note group=2> Il est essentiel d'indiquer la temp&eacute;rature et
     les autres conditions d'environnement dans lesquelles le
     ph&eacute;nom&egrave;ne est &eacute;tudi&eacute;. </note>

     <ptr type='relatedConcept' target='te04.06'>
     <admin depend=te84.11 type='domain'> plastics </admin>
     <ref type='bibliographic' depend=te84.11 target='ISO.472-1988'>
     p. 84 </ref>
     <admin depend=te84.11 type='creation' date='1988'
     resp='ISO/TC 61, Plastics'>
     </admin>
<termEntry>

<termEntry>

<!-- Referenced term entry: -->
<termEntry id=te04.06 n=te04.06>
     <term lang=en n=1> ageing </term>
     ...
     <term lang=fr n=2> vieillissement </term>
     ...
</termEntry>
```

Part IV

Additional Tag Sets

Chapter 14

Linking, Segmentation, and Alignment

This chapter discusses a number of ways in which encoders may represent analyses of the structure of a text which are not necessarily linear or hierarchic. In this chapter, tag sets and global attributes are provided for the following common requirements:

- to *link* disparate elements in a single document using the id attribute (section 14.1 ('Pointers') on p. 333);
- to link disparate elements in a single document without using the id attribute or to link elements in different documents (section 14.2 ('Extended Pointers') on p. 340);
- to *segment* text into elements convenient for the encoder and to mark arbitrary points within documents (section 14.3 ('Segments and Anchors') on p. 355);
- to represent *correspondence* or *alignment* among groups of text elements, both those with content and those which are empty (section 14.4 ('Correspondence and Alignment') on p. 358);[1]
- to *synchronize* elements of a text, that is to represent temporal correspondences and alignments among text elements (section 14.5 ('Synchronization') on p. 364) and also to align them with specific points in time (section xref target=SAsymp);
- to specify that one text element is *identical* to or a *copy* of another (section 14.6 ('Identical Elements and Virtual Copies') on p. 367);
- to *aggregate* possibly noncontinguous elements (section 14.7 ('Aggregation') on p. 369);
- to specify that different elements are *alternatives* to one another and to express *preferences* among the alternatives (section 14.8 ('Alternation') on p. 373);
- to associate segments of a text with interpretations or analyses of their significance (section 14.9 ('Connecting Analytic and Textual Markup') on p. 379).

These facilities all use the same basic set of techniques, which depend on the ability to point to an element which has some form of identifier. The most convenient such identifier, and that which is recommended by these Guidelines wherever possible, is provided by the global id attribute, as defined in section 3.5 ('Global Attributes') on p. 42. An extension to this mechanism is provided, for elements which are located in different SGML documents, or to which identifiers cannot be attached (perhaps because they are held on read-only media), known as the *TEI extended pointer mechanism* in section 14.2 ('Extended Pointers') on p. 340. For many of the topics discussed in this chapter, a choice of methods of encoding is offered, ranging from simple but less general ones, which use attribute values only, to more elaborate and more general ones, which use specialized elements.

The following DTD fragments show the overall organization of the additional tag set discussed in the remainder of this chapter. The file *teilink2.ent* begins by declaring a set of additional

[1] We use the term alignment as a special case of the more general notion of correspondence. Let A stand for an element with the attribute id=A, and suppose elements A1, A2 and A3 occur in that order and form one group, while elements B1, B2 and B3 occur in that order and form another group. Then a relation in which A1 corresponds to B1, A2 corresponds to B2 and A3 corresponds to B3 is an alignment. On the other hand, a relation in which A1 corresponds to B2, B1 to C2, and C1 to A2 is not an alignment.

attributes available globally when this tag set is enabled. This is followed by declarations for the attribute classes *pointer* and *pointerGroup* to which most of the elements discussed in this chapter belong; these attributes are all further described in the remainder of the chapter.

```
<!-- 14:  Global attributes for the TEI.linking tag set       -->
<!-- Text Encoding Initiative: Guidelines for Electronic      -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.    -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy    -->
<!-- in any form is granted, provided this notice is          -->
<!-- included in all copies.                                  -->

<!-- These materials may not be altered; modifications to     -->
<!-- these DTDs should be performed as specified in the       -->
<!-- Guidelines in chapter "Modifying the TEI DTD."           -->

<!-- These materials subject to revision. Current versions    -->
<!-- are available from the Text Encoding Initiative.         -->
<!-- When tag set TEI.linking is used, the following          -->
<!-- attributes may be attached to any element:               -->


<!ENTITY % a.linking '
          corresp            IDREFS              #IMPLIED
          synch              IDREFS              #IMPLIED
          sameAs             IDREF               #IMPLIED
          copyOf             IDREF               #IMPLIED
          next               IDREF               #IMPLIED
          prev               IDREF               #IMPLIED
          exclude            IDREFS              #IMPLIED
          select             IDREFS              #IMPLIED'     >

<!-- The following attributes apply to all pointer
elements: -->

<!ENTITY % a.pointer '
          type               CDATA               #IMPLIED
          resp               CDATA               #IMPLIED
          crdate             CDATA               #IMPLIED
          targType           NAMES               #IMPLIED
          targOrder          (Y | N | U)         U
          evaluate           (all | one | none)  #IMPLIED'     >

<!-- The following attributes apply to all pointer group      -->
<!-- elements:                                                -->

<!ENTITY % a.pointerGroup '  %a.pointer;
          domains            IDREFS              #IMPLIED
          targFunc           NMTOKENS            #IMPLIED'     >
```

The element declarations for this tag set are contained in the file *teilink2.dtd*:

```
<!-- 14:  Linking, Segmentation and Alignment                 -->
<!-- Text Encoding Initiative: Guidelines for Electronic      -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.    -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy    -->
<!-- in any form is granted, provided this notice is          -->
<!-- included in all copies.                                  -->

<!-- These materials may not be altered; modifications to     -->
<!-- these DTDs should be performed as specified in the       -->
<!-- Guidelines in chapter "Modifying the TEI DTD."           -->
```

```
<!-- These materials subject to revision. Current versions    -->
<!-- are available from the Text Encoding Initiative.         -->

<!-- ... declarations from section 14.1.3                     -->
<!--     (Links)                                              -->
<!--     go here ...                                          -->
<!-- ... declarations from section 14.2.1                     -->
<!--     (Extended pointers)                                  -->
<!--     go here ...                                          -->
<!-- ... declarations from section 14.3                       -->
<!--     (Segments and Anchors)                               -->
<!--     go here ...                                          -->
<!-- ... declarations from section 14.5.2                     -->
<!--     (Temporal specification)                             -->
<!--     go here ...                                          -->
<!-- ... declarations from section 14.7                       -->
<!--     (Aggregation)                                        -->
<!--     go here ...                                          -->
<!-- ... declarations from section 14.8                       -->
<!--     (Alternation)                                        -->
<!--     go here ...                                          -->
```

This tag set is made available by the mechanisms described in section 3.3 ('Invocation of the TEI DTD') on p. 39; this implies that the document type subset for a document using any of the tags or attributes described in this chapter must define a parameter entity *TEI.linking* with the value INCLUDE. For example, a document using this additional tag set and the prose base would begin with a series of declarations like the following:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
   <!ENTITY % TEI.prose   'INCLUDE'>
   <!ENTITY % TEI.linking 'INCLUDE' >
]>
```

14.1 Pointers

We say that one element *points* to others if the first has an attribute whose value is a reference to the others: such an element is called a *pointer element*, or simply a *pointer*. Among the pointers that have been introduced up to this point in these Guidelines are <note>, <ref> and <ptr>. These elements all indicate an association between one place in the document (the location of the pointer itself) and one or more others (the elements whose identifiers are specified by the pointer's target attribute). This element set defines a variation on this basic kind of pointer, known as a *link* which specifies both "ends" of an association. In addition, we define a syntax for representing locations in a document by a variety of means not dependent on the use of SGML identifiers.

14.1.1 Pointers and Links

In section 6.6 ('Simple Links and Cross References') on p. 147 we introduced the simplest pointer elements, <ptr> and <ref>. Here we introduce additionally the <link> element, which represents an association between two (or more) locations by specifying each location explicitly. Its own location is irrelevant to the intended linkage.

<ptr> defines a pointer to another location in the current document in terms of one or more identifiable elements. Attributes include:

target specifies the destination of the pointer as one or more SGML identifiers

<ref> defines a reference to another location in the current document, in terms of one or more identifiable elements, possibly modified by additional text or comment. Attributes include:

target specifies the destination of the reference as one or more SGML identifiers

<link> defines an association or hypertextual link among elements or passages, of some type not more precisely specifiable by other elements. Attributes include:

targets specifies the SGML identifiers of the elements or passages to be linked or associated.

The <ptr> element may be called a "pure pointer", because its primary function is simply to point. A pointer sets up a *connection* between an element (which, in the case of a pure pointer, can be thought of simply as a location in a document), and one or more others, known collectively as its *target*.

The <ptr> and <ref> elements bear a target attribute (in the singular), because they point, conceptually, at a single target, even if that target may be discontinuous in the document. The <link> element bears a targets attribute, with a plural name, because it specifies at least two targets, each of which is a unitary object. It may be thought of as a representing a *double link* between the objects specified.

As members of the class *pointer*, these elements share a common set of attributes:

type categorizes the pointer in some respect, using any convenient set of categories.

resp specifies the creator of the pointer.

crdate specifies when the pointer was created.

targType specifies the kinds of elements to which this pointer may point.

targOrder where more than one identifier is supplied as the value of the target attribute, this attribute specifies whether the order in which they are supplied is significant. Legal values are:

Y Yes: the order in which IDREFs are specified as the value of a target attribute should be followed when combining the targeted elements.

N No: the order in which IDREFs are specified as the value of a target attribute has no significance when combining the targeted elements.

U Unspecified: the order in which IDREFs are specified as the value of a target attribute may or may not be significant.

evaluate specifies the intended meaning when the target of a pointer is itself a pointer. Sample values include:

all if the element pointed to is itself a pointer, then the target of that pointer will be taken, and so on, until an element is found which is not a pointer.

one if the element pointed to is itself a pointer, then its target (whether a pointer or not) is taken as the target of this pointer.

none no further evaluation of targets is carried out beyond that needed to find the element specified in the pointer's target.

The targType and targOrder attributes may be used to constrain the scope of a link to certain element types. For example:

```
<link type='echo' targets='P1 P2'>
```

This is a complete unconstrained link, of type echo. It assumes only that there is an element with identifier P1 and another with identifier P2 somewhere in the current document.

```
<link type='echo' targType='p seg note' targets='P1 P2'>
```

This is a slightly more constrained link of the same type. P1 and P2 must now both identify a <p>, a <seg>, or a <note>, but there is no requirement as to which is which. (This may be useful if, as is often the case, different elements may participate in the same kind of link.)

```
<link type='echo' targType='p note' targOrder=Y targets='P1 P2'>
```

In this variation, not only must the link targets be either <p> or <note> elements, but the one with identifier P1 must be a <p>, and that with identifier P2 must be a <note>. Note that the present Guidelines provide no direct way of saying that P1 may identify either a <seg> or a <p> and P2 must identify a <note>.

These attributes are most useful if applied to a group of links, when additional constraints may also be specified, as further discussed in section 14.1.3 ('Groups of Links') on p. 337 below.

Double connection among elements could also be expressed by a combination of pointer elements, for example, two <ptr> elements, or one <ptr> element and one <note> element. All that is required is that the value of the target (or other pointing) attribute of the one be the value of the id attribute of the other. What the <link> element accomplishes is the handling of double connection by means of a single element. Thus, in the following encoding:

```
<ptr id=P1 target=P2> .....  <ptr id=P2 target=P1>
```

P1 points to P2, and P2 points to P1. This is logically equivalent to the more compact encoding:

```
<link targets='P1 P2'>
```

As noted above, all elements pointed to or linked by these elements must be identifiable using the global id attribute. This implies that they must be present in the same document, and that they must bear unique id values. Pointing or linking to external documents and pointing or linking where SGML identifiers are not available is implemented by the external pointing mechanisms discussed in section 14.2 ('Extended Pointers') on p. 340, where the <xptr> and <xref> elements are discussed. External links and links involving elements without identifiers do not require a special element; they may be represented using the standard <link> element, but an intermediate <xptr> element must be provided within the current document, to bear the id attribute used in the target of the link.

14.1.2 Using Pointers and Links

As an example of the use of these mechanisms which establish connections among elements, consider the practice (common in 18th century English verse and elsewhere) of providing footnotes citing parallel passages from classical authors. The figure shows the original page of Pope's *Dunciad* which is discussed in the text. Such footnotes can of course simply be encoded using the <note> element (see section 6.8 ('Notes, Annotation, and Indexing') on p. 152) without a target attribute, placed adjacent to the passage to which the note refers:[2]

```
<l>(Diff'rent our parties, but with equal grace</l>
<l>The Goddess smiles on Whig and Tory race,</l>
<note type='imitation' place=foot anchored=no>
    <bibl>Virg. &AElig;n. 10.</bibl>
    <quote><l>Tros Rutulusve fuat; nullo discrimine habebo.</l>
    <l>&mdash;&mdash; Rex Jupiter omnibus idem.</l>
    </quote>
</note>
<l>'Tis the same rope at sev'ral ends they twist,</l>
<l>To Dulness, Ridpath is as dear as Mist)</l>
```

This use of the <note> element can be called *implicit pointing* (or *implicit linking*). It relies on the juxtaposition of the note to the text being commented on for the connection to be understood. If it is felt that the mere juxtaposition of the note to the text does not make it sufficiently clear exactly what text segment is being commented on (for example, is it the immediately preceding line, or the immediately preceding two lines, or what?), or if it is decided to place the note at some distance from the text, then the pointing or the linking must be made explicit. We now consider various methods for doing that.

First, a <ptr> element might be placed at an appropriate point within the text to link it with the annotation:

```
<l>(Diff'rent our parties, but with equal grace</l>
<l>The Goddess smiles on Whig and Tory race,
<ptr rend='unmarked' target=N3.284></l>
<l>'Tis the same rope at sev'ral ends they twist,</l>
<l>To Dulness, Ridpath is as dear as Mist)</l>
<!-- elsewhere in the document ... -->
<note id=N3.284 type='imitation' place=foot anchored=no>
    <bibl>Virg. &AElig;n. 10.</bibl>
    <quote><l>Tros Rutulusve fuat; nullo discrimine habebo.</l>
    <l>&mdash;&mdash; Rex Jupiter omnibus idem.</l>
    </quote>
</note>
```

[2] The type attribute on the note is used to classify the notes using the typology established in the Advertisement to the work: "The *Imitations* of the Ancients are added, to gratify those who either never read, or may have forgotten them; together with some of the Parodies, and Allusions to the most excellent of the Moderns." In the source text, the text of the poem shares the page with two sets of notes, one headed "Remarks" and the other "Imitations".

The <note> element has been given an arbitrary identifier (N3.284) to enable it to be specified as the target of the pointer element. Because there is nothing in the text to signal the existence of the annotation, the rend attribute has been given the value unmarked.

Second, the target attribute of the <note> element can be used to point at its associated text, provided that an id attribute has been supplied for the associated text. Since, in this case, the note itself contains a pointer to the place in the text which it is annotating, this has also been encoded, using a <ref> element, which bears a target attribute of its own and contains a (slightly misquoted) extract from the text marked as a <quote> element:

```
<!--  ... -->
<note target=L3.284
    type='imitation' place=foot anchored=no>
    <ref target='L3.284' rend='sc'>
    Verse 283-84.
    <quote><l>&mdash;&mdash;. With equal grace
           <l>Our Goddess smiles on Whig and Tory race.</l>
    </quote></ref>
    <bibl>Virg. &AElig;n. 10.</bibl>
    <quote>
    <l>Tros Rutulusve fuat; nullo discrimine habebo.</l>
    <l>&mdash;&mdash; Rex Jupiter omnibus idem.</l>
    </quote>
</note>
```

Combining these two solutions gives us the following associations:

- a pointer within one line indicates the note
- the note indicates the line
- a pointer within the note indicates the line

Note that we do not have any way of pointing from the line itself to the note: the association is implied by containment of the pointer. We do not as yet have a true double link between text and note.

Thirdly, therefore, we supply identifiers for both verse line and annotation, and use a <link> element to associate the two. Note that the <ptr> element and the target attribute on the <note> may now be dispensed with:

```
<l id=L3.283>(Diff'rent our parties, but with equal grace</l>
<l id=L3.284>The Goddess smiles on Whig and Tory race,</l>
<l id=L3.285>'Tis the same rope at sev'ral ends they twist,</l>
<l id=L3.286>To Dulness, Ridpath is as dear as Mist)</l>
<!-- elsewhere in the document ... -->
<note id=N3.284
    type='imitation' place=foot anchored=no>
    <ref target='L3.284' rend='sc'>
    Verse 283-84.
    <quote><l>&mdash;&mdash;. With equal grace</l>
           <l>Our Goddess smiles on Whig and Tory race.</l>]</ref>
    <bibl>Virg. &AElig;n. 10.</bibl>
    <quote><l>Tros Rutulusve fuat; nullo discrimine habebo.</l>
    <l>&mdash;&mdash; Rex Jupiter omnibus idem.</l>
    </quote>
</note>
<!-- ... and yet elsewhere in the document ... -->
<link targType='note l' targOrder=Y targets='N3.284 L3.284'>
```

The targets attribute of the <link> element here bears the identifiers of the note followed by that of the verse line. The targType and targOrder attributes may be used to enable application programs to check that the identifiers in fact pick out a <note> element and an <l> element and in that order. If targOrder has the value N, then the elements indicated by the targets attribute have to be either <note> or <l> elements, but are otherwise unconstrained. If neither attribute is present, then the only constraint is that the identifiers given must apply to some element within the current document.

For completeness, we could also allocate an identifier to the reference within the note and encode the association between it and the verse line in the same way:

```
<!-- ... -->
<note id=N3.284 type='imitation' place=foot anchored=no>
    <ref id=R3.284 rend='sc'>
    <!-- ... -->
    </ref>
</note>
<!-- ... -->
<link targType='ref  l' targOrder=Y targets='R3.284 L3.284'>
```

Indeed, the two <link>s could be combined into one, as follows:

```
<link targType='note ref l' targOrder=Y
      targets='N3.284 R3.284 L3.284'>
```

14.1.3 Groups of Links

Clearly, there are many reasons for which an encoder might wish to represent a link or association between different elements. For some of them, specific elements are provided in these Guidelines; some of these are discussed elsewhere in the present chapter. The <link> element is a general purpose element which may be used for any kind of association. The element <linkGrp> may be used to group links of a particular type together in a single part of the document; such a collection may be used to represent what is sometimes referred to in the literature of Hypertext as a *web*, a term introduced by the Brown University FRESS project in 1969.

<linkGrp> defines a collection of associations or hypertextual links.

As a member of the class *pointerGroup*, this element shares the following attributes with other members of that class:

domains optionally specifies the identifiers of the elements within which all elements indicated by the contents of this element lie.

targFunc describes the function of each of the values of the targets attribute of the enclosed <link>, <join> or <alt> tags.

It is also a member of the *pointer* class, and therefore also carries the attributes specified in section 14.1.1 ('Pointers and Links') on p. 333 above, in particular the type attribute:

type categorizes the pointer in some respect, using any convenient set of categories.

The <linkGrp> element provides a convenient way of establishing a default for the type attribute on a group of links of the same type: by default, the type attribute on a <link> element has the same value as that given for type on the enclosing <linkGrp>.

Typical software might hide a web entirely from the user, but use it as a source of information about links, which are displayed independently at their referenced locations. Alternatively, software might provide a direct view of the link collection, along with added functions for manipulating the collection, as by filtering, sorting, and so on. To continue our previous example, this text contains many other notes, of a kind similar to the one shown above. To avoid having to repeat the type=imitation on each <note>, we may specify it once for all on a <linkGrp> element containing all links of this type. The targType and targOrder attributes can also be specified for a <linkGrp> element:

```
<l id=L2.79>A place there is, betwixt earth, air and seas</l>
<l id=L2.80>Where from Ambrosia, Jove retires for ease.</l>
<l> ...
<l id=L2.88>Sign'd with that Ichor which from Gods distills.</l>
<l> ...
<l id=L3.283>(Diff'rent our parties, but with equal grace</l>
<l id=L3.284>The Goddess smiles on Whig and Tory race,</l>
<l id=L3.285>'Tis the same rope at sev'ral ends they twist,</l>
<l id=L3.286>To Dulness, Ridpath is as dear as Mist)</l>
<!-- ... -->
<!-- elsewhere in the document ... -->
```

```
<note id=N2.79 place=foot anchored=no><bibl>Ovid Met. 12.</bibl>
    <quote lang=la>
        <l>Orbe locus media est, inter terrasq; fretumq;
        <l>C&oelig;lestesq; plagas &mdash;
    </quote>
</note>
<note id=N2.88 place=foot anchored=no>
    Alludes to <bibl>Homer, Iliad 5</bibl>
<!-- .... -->
</note>
<!-- ... -->
<note id=N3.284 place=foot anchored=no><bibl>Virg. &AElig;n. 10.</bibl>
    <quote><l>Tros Rutulusve fuat; nullo discrimine habebo.</l>
    <l>&mdash;&mdash; Rex Jupiter omnibus idem.</l>
    </quote>
</note>
<!-- ... -->
<!-- yet elsewhere in the document ... -->
<linkGrp type='imitation' targType='note l' targOrder=Y>
  <link targets='N2.79 L2.79'>
  <link targets='N2.88 L2.88'>
  <!-- ... -->
  <link targets='N3.284 L3.284'>
  <!-- ... -->
</linkGrp>
```

Additional information for applications that use <linkGrp> elements can be provided by means of special attributes. First, the domains attribute can be used to identify the text elements within which the individual targets of the links are to be found. Suppose that the text under discussion is organized into a <body> element, containing the text of the poem, and a <back> element containing the notes. Then the domains attribute can have as its value the identifiers of the <body> and the <back>, to enable an application to verify that the link targets are in fact contained by appropriate elements, or to limit its search space:

```
<body id=dunciad>
<!-- .... -->
<l id=L2.79>A place there is, betwixt earth, air and seas</l>
<l id=L2.80>Where from Ambrosia, Jove retires for ease.</l>
<l> ...
<l id=L2.88>Sign'd with that Ichor which from Gods distills.</l>
<l> ...
<l id=L3.283>(Diff'rent our parties, but with equal grace</l>
<l id=L3.284>The Goddess smiles on Whig and Tory race,</l>
<l id=L3.285>'Tis the same rope at sev'ral ends they twist,</l>
<l id=L3.286>To Dulness, Ridpath is as dear as Mist)</l>
<!-- ... -->
</body>
<back id=dunnotes>
<!-- ... -->
<note id=N2.79 place=foot anchored=no><bibl>Ovid Met. 12.</bibl>
    <quote lang=la>
        <l>Orbe locus media est, inter terrasq; fretumq;
        <l>C&oelig;lestesq; plagas &mdash;
    </quote>
</note>
<note id=N2.88 place=foot anchored=no>
    Alludes to <bibl>Homer, Iliad 5</bibl>
<!-- .... -->
</note>
<!-- ... -->
<note id=N3.284 place=foot anchored=no><bibl>Virg. &AElig;n. 10.</bibl>
```

```
    <quote><l>Tros Rutulusve fuat; nullo discrimine habebo.</l>
    <l>&mdash;&mdash; Rex Jupiter omnibus idem.</l>
    </quote>
</note>
<!-- ... -->
</back>
<!-- elsewhere in the document ... -->
<linkGrp type='imitation'
         domains='dunciad dunnotes'
         targType='note l'
         targOrder=Y>
  <link targets='N2.79 L2.79'>
  <link targets='N2.88 L2.88'>
  <!-- ... -->
  <link targets='N3.284 L3.284'>
  <!-- ... -->
</linkGrp>
```

Note that there must be a single parent element for each "domain"; if some notes are contained by a section with identifier `dunnotes`, and others by a section with identifier `dunimits`, an intermediate pointer must be provided (as described in section 14.1.4 ('Intermediate Pointers') on p. 340) within the <linkGrp> and its identifier used instead.

Next, the targFunc attribute can be used to provide further information about the role or function of the various targets specified for each link in the group.

The value of the targFunc attribute is a list of names (formally, SGML name tokens), one for each of the targets in the link; these names can be chosen freely by the encoder, but their significance should be documented in the encoding declaration in the header.[3] In the current example, we might think of the note as containing the *source* of the imitation and the verse line as containing the *goal* of the imitation. Accordingly, we can specify the <linkGrp> in the preceding example thus:

```
<linkGrp type='imitation'
         domains='dunciad dunnotes'
         targType='note l'
         targFunc='source goal'
         targOrder=Y>
  <link targets='N2.79 L2.79'>
  <link targets='N2.88 L2.88'>
  <!-- ... -->
  <link targets='N3.284 L3.284'>
  <!-- ... -->
</linkGrp>
```

The <link> and <linkGrp> elements are formally defined as follows:

```
<!-- 14.1.3:  Links                                          -->
<!ELEMENT link          - o  EMPTY                            >
<!ATTLIST link               %a.global;
          resp               CDATA               #IMPLIED
          crdate             CDATA               #IMPLIED
          targType           NAMES               #IMPLIED
          targOrder          (Y | N | U)         U
          evaluate           (all | one | none)  #IMPLIED
          targets            IDREFS              #REQUIRED
          type               CDATA               %INHERITED;    >
<!ELEMENT linkGrp       - -  (link | ptr | xptr)+               >
<!ATTLIST linkGrp            %a.global;
                             %a.pointerGroup;                   >
<!-- This fragment is used in sec. 14                          -->
```

[3] No special element is provided for this purpose at present: the information should be supplied as a series of paragraphs at the end of the <encodingDesc> element described in section 5.3 ('The Encoding Description') on p. 93.

14.1.4 Intermediate Pointers

In the preceding examples, we have shown various ways of linking an annotation and a single verse line. However, the example cited in fact requires us to encode an association between the note and a *pair* of verse lines (lines 284 and 285).

There are a number of possible ways of correcting this error: one could use the target and targetEnd attributes of the <note> element to delimit the span to which the note applies (see further section 6.8 ('Notes, Annotation, and Indexing') on p. 152). Alternatively one could create an element to encode the couplet itself and assign it an id attribute, which can then be linked to the <note> and <ref> elements. This could be done either explicitly by means of a <lg> element, as defined in section 6.11.1 ('Core Tags for Verse') on p. 177, or a <seg> element, as defined in section 14.3 ('Segments and Anchors') on p. 355, or implicitly, by means of the <join> element discussed in section 14.7 ('Aggregation') on p. 369.

A third possibility however, is to use an "intermediate pointer" as follows:

```
<l id=L3.283>(Diff'rent our parties, but with equal grace</l>
<l id=L3.284>The Goddess smiles on Whig and Tory race,</l>
<!-- ... -->
<ptr id=L3.283284 target='L3.283 L3.284' targOrder=Y >
```

When the target attribute of a <ptr> or <ref> element specifies more than one element, the indicated elements are always understood to be combined or aggregated in some way to produce the object of the pointer.

In this example, the targOrder attribute should be specified to indicate that the order in which identifier values are supplied in the target attribute is significant. The id attribute provides an identifier which can then be linked to the <note> and <ref> elements:

```
<link targets='N3.284 R3.284 L3.283284'
      targType='note ref ptr' evaluate=all>
```

The evaluate=all attribute value is used on the <link> element to specify that any pointer encountered as a target of that element is itself evaluated. If evaluate had the value none, the link target would be the pointer itself, rather than the objects it points to.

Where a <linkGrp> element is used to group a collection of <link> elements, any intermediate pointer elements used by those <link> elements should be included within the <linkGrp>.

Intermediate pointers of this kind are particularly important when extended pointers (discussed in the next section) are in use.

14.2 Extended Pointers

Where the object of a link or pointer element is not contained within the current document, or where it does not bear an id attribute, it is not possible to point at it with a <ptr> or <ref> element, nor to link it directly with a <link> element, because no IDREF value can be supplied for the target or targets attribute of these elements. In such cases, the encoder must indicate the intended element indirectly by means of the elements discussed in this section. These elements identify their target using a special TEI-defined extended pointer notation, defined in section 14.2.2 ('Extended Pointer Syntax') on p. 341 below and designed for compatibility with HyTime. [4]

14.2.1 Extended Pointer Elements

To point or refer to locations in the current or some other document *without* requiring that the target bear an SGML identifier, the following elements should be used:

<xptr> defines a pointer to another location in the current document or an external document.

[4] HyTime is an international standard (ISO 10744) built on SGML. It provides facilities for representing both static and dynamic information for processing and interchange by hypertext and multimedia applications. See *ISO/IEC 10744 Information Technology — Hypermedia/Time-based Structuring Language (HyTime)* ([Geneva]: International Organization for Standardization, 1992).

<xref> defines a reference to another location in the current document, or an external document, using an extended pointer notation, possibly modified by additional text or comment.

These elements are both members of the element class *pointer*, and therefore carry the same attributes as other members of that class, listed above (see section 14.1.1 ('Pointers and Links') on p. 333).

They are also members of the class *xPointer*, from which they inherit the following attributes:

doc specifies the document within which the desired location is to be found.

from specifies the start of the destination of the pointer, as an expression in the TEI extended-pointer notation described in section 14.2 ('Extended Pointers') on p. 340.

to specifies the endpoint of the destination of the pointer, as an expression in the TEI extended pointer notation.

Unlike the pointer elements discussed in the previous section, these elements do not specify their target by means of a target attribute. Instead these elements use one or both of the attributes from and to to delimit a portion of some document specified by the doc attribute. In all other respects, these elements correspond with the elements <ptr> and <ref> discussed in sections 6.6 ('Simple Links and Cross References') on p. 147, and 14.1 ('Pointers') on p. 333. Note that there is no element <xlink> corresponding with the <link> element; links can be made both within and between documents using the same syntax, as further discussed below.

The values of the from and to attributes on the <xptr> and <xref> elements indicate the point or passage being referred to by showing how to locate it, using one or more special keywords, as defined below in section 14.2.1 ('Extended Pointer Elements') on p. 340. Examples are given there.

The <xptr> and <xref> elements are formally defined as follows:

```
<!-- 14.2.1:  Extended pointers                                  -->
<!ELEMENT xref          - O  (%paraContent)                       >
<!ATTLIST xref               %a.global;
                             %a.xPointer;                         >
<!ELEMENT xptr          - O  EMPTY                                >
<!ATTLIST xptr               %a.global;
                             %a.xPointer;                         >
<!-- This fragment is used in sec. 14                            -->
```

14.2.2 Extended Pointer Syntax

As noted above, the elements <xptr> and <xref> are used to represent a link between their own location (the "link origin") and some other location (the "destination"), which may or may not be in the same document. Software supporting intra- and inter-document links (e.g. hypertext systems) should provide access from the location of such an element to the destination.

This section defines the allowable values for the attributes from, to, and doc of the <xptr> and <xref> elements.

An <xptr> or <xref> element with *no* attributes at all is, by definition, a link to the root (i.e. the *document element* — by default, this is the <TEI.2> element) of the document in which it appears.

The doc attribute value must be the name of an entity declared in the SGML document type declaration. If only the doc attribute is given a value, then by definition the destination is the entire entity named by the doc value. A more specific location within another entity must be specified with the from and the to attributes, as described below.

The from and the to attributes indicate the specific location pointed at, within the entity named by the doc attribute (or within the current document, if no doc value is given). Their values are referred to below as *location pointer specification*s. When both attributes are specified, the span pointed at by the element runs from the starting point of the span indicated by from to the ending point of the string specified by to. If the latter precedes the former in the document, then the pointer is in error and fails. If only the from attribute is specified, the to attribute defaults to the same value; the effect is that the element as a whole points to the span indicated by the from attribute. It is a semantic error to specify a value for to but not for from.

14.2.2.1 Location Ladders

Each location pointer specification consists of a sequence of *location term*s, each of which consists of a keyword specifying a *location type* followed by one or more parenthesized parameter lists, each of which specifies a *location value* via a list of parameters. Location types and values, and the parameters within a location value, must be separated by white space characters.

Using terms borrowed from *HyTime*, we say that each TEI location term in a specification provides the *location source* for the next, and the entire specification is equivalent to a *location ladder*. By specifying the entire ladder in a single attribute value, the TEI extended pointer mechanism greatly reduces the syntactic and processing complexity of hypertextual pointers.

In formal terms:[5]

```
ladder   ::=  locterm
          |   ladder locterm
```

14.2.2.2 Location Terms

The keywords used in location terms are these; references to "the tree" mean the tree representing the SGML document hierarchy.

root points at the root of the target document
here points at the location of the pointer
id points at an ID within the target document
ref gives a "canonical reference" to a location in the target document
child indicates an element found by descending one level in the tree
descendant indicates an element found by descending one or more levels in the tree
ancestor indicates an element found by ascending one or more levels in the tree
previous indicates an element found by traversing the older siblings of the current location source
next indicates an element found by traversing the younger siblings of the current location source
preceding indicates an element found by traversing the entire portion of the document preceding the current location source
following indicates an element found by traversing the entire portion of the document which follows the current location source
pattern specifies a regular expression to be located within the existing location source
token points at one or more *tokens* in the character content of the location source
str points at one or more characters in the character content of the location source
space points at a location using coordinates in some (application-defined) n-dimensional space
foreign points at a location using some non-SGML method, and gives the name of the method
HyQ points at a location using the HyQ query language defined by ISO 10744 (HyTime)
ditto (in the to attribute only) points at the same span as was indicated by the from attribute

In formal terms:

```
locterm ::=   'ROOT'                        // default first location
          |   'HERE'                        // location of the xptr
          |   'ID' '(' NAME ')'             // only one ID allowed.
          |   'REF' '(' characters ')'      // only one ref allowed
          |   'CHILD' steps
          |   'DESCENDANT' steps
          |   'ANCESTOR' steps
          |   'PREVIOUS' steps
          |   'NEXT' steps
          |   'PRECEDING' steps
          |   'FOLLOWING' steps
          |   'PATTERN' regs                // mult patterns allowed
          |   'TOKEN' '(' range ')'
```

[5] The notation used for this formal grammar is that defined in chapter 39 ('Formal Grammar for the TEI-Interchange-Format Subset of SGML') on p. 993.

```
|    'STR' '(' range ')'
|    'SPACE' '(' NAME ')' pointpair
|    'FOREIGN' parms
|    'HYQ' parms
|    'DITTO'                          // valid only in TO att.
```

Note that the keywords, though shown here quoted in uppercase, are not case sensitive.

Each location term specifies a location in the target document; this location may be a single point, more often a span of text (often the span of a single element) within the target document. The location ladder as a whole is interpreted from left to right, and each location term specifies a location relative to the location specified by the sequence prior to that point (i.e. to its location source). Unless `here` or `id` is specified as the first location term, the beginning location source is always `root`. An empty location sequence thus is the same as `root` and specifies the entire destination entity.

In general, the search for the location specified by a location term will be conducted only within its location source (i.e. within the location already identified by preceding location terms). There are however several exceptions. The terms `root`, `here`, and `id` all ignore the location source defined by any preceding terms and therefore make sense only as the first items in the ladder. The terms `ancestor`, `next`, and `previous` do not ignore the location source, but select a new span from the adjacent or enclosing portions of the text, and not from within the location source. Finally the location terms `foreign`, `space`, and `HyQ` are not defined fully here; they may or may not ignore the existing location source.

Some of the location terms make sense only in SGML documents; these are `id`, `child`, `ancestor`, `descendant`, `previous`, `next`, `preceding`, and `following`. The latter six involve traversing the tree representing the SGML document hierarchy and are most easily understood when their location source is a single SGML element. If the location source is not a single SGML element, the tree-traversal keywords operate upon its beginning end-point, its "front end" (in English, this will be the leftmost point of the location source; in Arabic or Hebrew it will be the rightmost point). In this case `child` and `descendant` have no meaning, since character data has no descendants in the document tree; the first `ancestor` of such a location source is the element immediately containing the character data in question, and the siblings referred to by `next` and `previous` are the other children of that immediately containing element.

The details of each keyword are given below, along with definitions of their syntax and semantics of their results. Examples are also provided. It is strongly recommended that when IDs are available, they should be used in preference to the other methods for pointing defined here.

For all keywords, the description assumes that the target document does in fact contain a span or element which matches the description; otherwise, the location term has no referent and is said to "fail". If any location term fails, the entire pointer fails. No backtracking or retrying is performed (and indeed for the most part the location terms are defined as having only one matching location, so backtracking would in most cases lead to no better result).

14.2.2.3 The ROOT Keyword

The location term `root` selects the root of the destination document tree; in SGML terms, this is the "document element". Since it ignores any existing location source, the `root` keyword makes sense only as the first location term in the ladder. Since `root` is assumed as the implicit first term in any ladder, the following two location ladders have the same meaning:

```
ROOT DESCENDANT (2 DIV1) DESCENDANT (2 DIV1)
```

14.2.2.4 The HERE Keyword

The keyword `here` designates the location at which the pointer element itself is situated; it allows extended pointers to select items like "the paragraph immediately preceding the one within which this pointer occurs". Since it ignores any existing location source, this keyword typically makes sense only as the first location term in a location specification.

To designate "the paragraph preceding the current one", the following location ladder could be used:

```
HERE ANCESTOR (1 P) PREVIOUS (1 P)
```

(See below for descriptions of the keywords `ancestor` and `previous`.)

14.2.2.5 The ID Keyword

The resulting location is the element within the destination entity whose ID attribute has the value specified as the location value. The ID location type typically makes sense only as the *first* location pair in a location specification, but there is no syntactic requirement that it be so.

For example, the location specification

```
ID (a27)
```

chooses the necessarily unique element of the destination entity which has an attribute of declared value of type ID, whose value is `a27`.

14.2.2.6 The REF Keyword

The resulting location is an element which can be found by interpreting the location value in accordance with document-specific rules for a *canonical reference*. Such reference systems, particularly common in documents of interest to classical and biblical scholars, must also be defined in the TEI header, using the `<refsDecl>` element (see section 5.3.5 ('The Reference System Declaration ') on p. 100). If more than one element matches the canonical reference, the first one encountered is chosen.

For example, the location specification

```
REF (MT.2.1)
```

chooses the first element of the destination entity which is identified by the canonical reference 'MT.2.1'

14.2.2.7 The CHILD Keyword

The `child` location type specifies an element or span of character data in the document hierarchy using a location value which functions as a *domain-style* address. The value is a series of parenthesized steps, separated by white space. Each such step represents one level of the hierarchy within the location source. Each step may contain one or more parameters separated by white space and interpreted in order as follows:

1. an instance indicator, which is a signed or unsigned integer or the special value `ALL`
2. optionally, an expression matching an SGML generic identifier
3. optionally, one or more pairs of expressions, the first matching an SGML attribute name and the second matching an SGML attribute value

In formal terms, the location value of `child` is a series of *steps*:

```
steps    ::=  '(' step ')'
          |   steps '(' step ')'

step     ::=  instance
          |   instance element
          |   instance element avspecs

avspecs  ::=  attribute value
          |   avspecs attribute value
```

Location values of the same form are also used by the keywords `descendant`, `ancestor`, `previous`, and `next`; details of the interpretation may vary from keyword to keyword.

If an instance indicator alone is specified, as a number *n*, it selects the *n*th child of the location source. If the special value `ALL` is given, then *all* the children of the location source are selected. If the instance indicator is specified with following parameters, it selects all, or the *n*th, among

those children of the location source which satisfy the other parameters. If a negative number is given, the *n*th child is counted from the last child of the location source to the first. The location source must contain at least *n* children;[6] if it does not, the `child` term fails.

In formal terms, the first parameter of a *step* is an instance indicator, which in turn is either the special value `ALL` or a signed integer:

```
instance ::=  'ALL'
          |   signed

signed   ::=  NUMBER                          // default sign is +
          |   '+' NUMBER
          |   '-' NUMBER
```

If a second parameter is given, it is interpreted as an SGML generic identifier, and only elements of the type indicated will be selected. For example, the location specification

```
CHILD (3 DIV1) (4 DIV2) (29 P)
```

chooses the 29th paragraph of the fourth sub-division of the third major division of the initial location source. The location specification

```
CHILD (3 DIV1) (4 DIV2) (-2 P)
```

chooses the next-to-last paragraph of the fourth <div2> of the third <div1> in the location source.

Constraint by generic identifier is strongly recommended, because it makes links more perspicuous and more robust. It is perspicuous because humans typically refer to things by type: as "the second section", "the third paragraph", etc. It is robust because it increases the chance of detecting breakage if (due to document editing) the target originally pointed at no longer exists.

The generic identifier may be specified as a normal SGML name, as a (parenthesized) regular expression, or using the reserved values `#CDATA` or `*`. Regular expressions take the form described below; the location term

```
CHILD (3 (DIV[123])
```

matches the third element which has a generic identifier of `div1`, `div2`, or `div3`. If the generic identifier is specified as `*`, any generic identifier is matched; this means that "CHILD (2 *)" is synonymous with `CHILD (2)`. If the second parameter is `#CDATA`, the location term selects only untagged sub-portions of an element having SGML mixed content.

The location ladder

```
CHILD (3 #CDATA)
```

thus chooses the third span of character data directly contained by the current location source. If the location source is a paragraph containing

1. a sentence (A)
2. an embedded quotation, marked as a <q>
3. another sentence (B)
4. an embedded note, marked as a <note>
5. another sentence (C)
6. a second embedded quotation, marked as a <q>

where the three sentences A, B, and C are character data enclosed by no element smaller than the paragraph itself, then `CHILD (3 #CDATA)` selects sentence C, while `CHILD (3)` selects sentence B.

If specified as a name (i.e. without parentheses), the generic identifier is case sensitive if and only if the SGML declaration specifies that generic identifiers are case sensitive (by default they are not). If specified as a regular expression, the expression given is always case sensitive; in the usual case this means the regular expression should be in uppercase, as in the examples here.

In formal terms the second parameter of a *step* is defined thus:

[6] Strictly speaking, |n| (absolute value of n) children.

```
element  ::=  NAME
          |    '#CDATA'
          |    '*'
          |    '(' regular ')'
```

The third and fourth parameters, if given, are interpreted as an attribute-value pair, and only elements which match that pair in the way described below will be selected; the fourth and fifth parameters, and all following pairs of parameters, are interpreted in the same way. When more than one pair is given, all must be matched.

The third, fifth, seventh, etc., parameters are interpreted, if specified, as attribute names. Like generic identifiers, attribute names may be specified as * in location ladders in the (unlikely) event that an attribute value constitutes a constraint regardless of what attribute name it is a value for. The attribute name parameter may also be specified as a parenthesized regular expression.

For example, the location term

```
CHILD (1 * TARGET *)
```

selects the first child of the location source for which the attribute target has a value. The location term

```
CHILD (1 * (TARGET(S?)) *)
```

will select the first child of the location source for which an attribute called either target or targets has a value.

As with generic identifiers, attribute names are case sensitive if and only if the SGML declaration says they are; regular expressions are always case sensitive and should usually be uppercased, as shown here.

In formal terms, the attribute-name parameter of a tree-traversal step is defined thus:

```
attribute ::= NAME
          |    '*'
          |    '(' regular ')'
```

If a fourth, sixth, eighth, etc., parameter is specified, it is interpreted as an attribute value, and only elements satisfying the other constraints and also bearing an attribute of the specified name and value will be selected. The attribute value may be specified exactly as in an SGML document; as a consequence, if the attribute value to be specified contains white space characters, it must be enclosed in quotation marks. The attribute value may also be specified as a regular expression, enclosed in parentheses, or using the two special values #IMPLIED and *.

For example, the location specification

```
CHILD (1 * N 2) (1 * N 1)
```

chooses an element using the global n attribute. Beginning at the location source, the first child (whatever kind of element it is) with an n attribute having the value 2 is chosen; then that element's first direct sub-element having the value 1 for the same attribute is chosen.

The location specification

```
CHILD (1 FS RESP ((lanc|LANC)(s|S|ashire|ASHIRE)))
```

selects the first child of the location source which is an <fs> element bearing a resp attribute with the value lancs, lancashire, LANCS, or LANCASHIRE (as well as other possible combinations which are left to the reader's ingenuity). If specified with quotation marks or as a regular expression, the attribute-value parameter is case-sensitive; otherwise not.

The location specification

```
CHILD (1 FS RESP #IMPLIED)
```

selects the first child of the location source which is an <fs> element for which the resp attribute has been left unspecified. The location ladder

```
ROOT DESCENDANT (1 (DIV[01234567]) TYPE chapter N 2)
```

selects the second chapter of a text, regardless of whether chapters are tagged using , <div1>, <div2>, or some other text-division element. It does so by selecting the first text-division element in the document which is of type chapter and has the n value 2.

In formal terms, the attribute-value parameter of a tree-traversal step is defined thus:

```
value    ::=  LITERAL                  // i.e. quoted string.
          |   NAME                     // As for attribute values in
          |   NUMBER                   // document, NMTOKENs need not
          |   NUMTOKEN                 // be quoted
          |   '#IMPLIED'               // No value specified, no default
          |   '*'                      // Any value matches.
          |   '(' regular ')'
```

14.2.2.8 The DESCENDANT Keyword

If the descendant keyword is used, the location term selects an element or character-data string which is a descendant of the current location source. Like child, descendant takes as a value a series of one or more parenthesized steps, which may contain the same four parameters described above. The set of elements and strings which may be selected, however, is the set of all descendants of the location source (i.e. the set of all elements contained by it), rather than only the set of immediate children.

The location specification

```
ID (a23) DESCENDANT (2 TERM LANG DE)
```

thus selects the second <term> element with a lang of de occurring within the element with an id of a23. The search for matching elements occurs in the same order as the SGML data stream; in terms of the document tree, this amounts to a depth-first left-to-right search.

If the instance number is negative, the search is a depth-first right-to-left search, in which the right-most, deepest matching element is numbered -1, etc. The location specification

```
DESCENDANT (-1 NOTE)
```

thus chooses the last <note> element in the document, that is, the one with the rightmost start-tag.

14.2.2.9 The ANCESTOR Keyword

The ancestor location term selects an element from among the direct ancestors of the location source in the document hierarchy. The location value is of the same form as defined for the child and descendant location types. However, the ancestor keyword selects elements from the list of containing elements or "ancestors" of the location source, counting upwards from the parent of the location source (which is ancestor number 1) to the root of the document instance (which is ancestor number -1).

The location source must have at least as many ancestors as the absolute value of the instance number specified as the first parameter of the step. The ancestor type thus may not be specified as the first component of a location specification, because the initial location source in effect at that point is the root, which has no ancestors.

For example, the location term

```
ANCESTOR (1 * N 1) (1 DIV)
```

first chooses the smallest element properly containing the location source and having attribute n with value 1; and then the smallest element properly containing it. The location term

```
ANCESTOR (1)
```

chooses the immediate parent of the location source, regardless of its type or attributes. The location term

```
ANCESTOR (1 * LANG fr)
```

selects the smallest ancestor for which the lang attribute has the value fr. The term

```
ANCESTOR (-1 * LANG fr)
```

selects the *largest* ancestor for which the lang attribute has the value fr. Without the attribute specification, the term

```
ANCESTOR (-1)
```

selects the largest ancestor of the location source and is thus normally synonymous with the keyword ROOT. If the instance indicator is given as ALL, then all the ancestor elements which match the later parameters are selected; since the largest of these will necessarily include all the others, the value ALL is thus synonymous with the value (-1) when used with ANCESTOR.

Finally, the term

```
ANCESTOR (1 (DIV[0123456789]?))
```

chooses the smallest element of any level which contains the location source.

14.2.2.10 The PREVIOUS Keyword

The previous keyword selects an element or character-data string from among those which precede the location source within the same containing element. We speak of the elements and character-data strings contained by the same parent element as *siblings*; those which precede a given element or string in the document are its *elder siblings*; those which follow it are its *younger siblings*.

The instance number in the location value of a previous term designates the nth elder sibling of the location source, counting from most recent to less recent. The location ladder

```
ID (a23) PREVIOUS (1)
```

thus designates the element immediately preceding the element with an id of a23. Negative instance numbers also designate elder siblings, counting from the eldest sibling to the youngest. The location source must have at least as many elder siblings as the absolute value of the instance number. If the location source has at least one elder sibling, then the location term

```
PREVIOUS (-1)
```

designates its eldest sibling and is thus synonymous with the ladder

```
ANCESTOR (1) CHILD (1)
```

The value ALL may be used to select the entire range of elder siblings of an element: the location ladder

```
ID (a23) PREVIOUS (ALL)
```

thus designates the set of elements which precede the element with an id of a23 and are contained by the same parent.

14.2.2.11 The NEXT Keyword

The keyword next behaves like previous, but selects from the younger siblings of the location source, not the elder siblings. The location ladder

```
ID (a23) NEXT (1)
```

thus designates the element or string immediately following the element which has an id of a23. Negative instance numbers also designate younger siblings, counting from the youngest sibling to the location source. The location source must have at least as many younger siblings as the absolute value of the instance number. If the location source has at least one younger sibling, then the location term

```
NEXT (-1)
```

designates its youngest sibling and is thus synonymous with the ladder

```
ANCESTOR (1) CHILD (-1)
```

14.2.2.12 The PRECEDING Keyword

The preceding keyword selects an element or character-data string from among those which precede the location source, without being limited to the same containing element. The set of elements and strings which may be selected is the set of all elements and strings in the entire document which occur or begin before the location source. (For purposes of the keywords PRECEDING and FOLLOWING, elements are interpreted as occurring where their start-tag

occurs.) The PRECEDING keyword thus resembles PREVIOUS but differs in searching a larger set of strings and elements; its result is not guaranteed to be a subset of its location source.

The instance number in the location value of a preceding term designates the nth element or character-data string preceding the location source, counting from most recent to less recent. The location ladder

```
ID (a23) PRECEDING (5)
```

thus designates the fifth element or string before the element with an id of a23. Negative instance numbers also designate preceding elements or strings, counting from the eldest to the youngest; the ladder

```
ID (a23) PRECEDING (-5)
```

thus selects the fifth element or string in the document overall, assuming that it precedes the element with an id of a23. It is thus normally synonymous with

```
ROOT DESCENDANT (5)
```

differing only in that it fails if four items or fewer precede element A23.

The location source must have at least as many elder siblings as the absolute value of the instance number; otherwise, the preceding term fails.

The value ALL may be used to select the entire portion of the document preceding the beginning of the location source: the location ladder

```
ID (a23) PRECEDING (ALL)
```

designates the entire portion of the document preceding the start-tag for element A23.

14.2.2.13 The FOLLOWING Keyword

The keyword following behaves like preceding, but selects from the portion of the document following the location source, not that preceding it. The location ladder

```
ID (a23) FOLLOWING (1)
```

thus designates the element or string immediately following the element which has an id of a23. Negative instance numbers select elements or strings counting from the end of the document to the location source. There must be at least as many elements or strings following the location source as the absolute value of the instance number. If the location source has at least one following element or string, then the location term

```
FOLLOWING (-1)
```

designates the youngest of these and is thus synonymous with the ladder

```
ROOT DESCENDANT (-1)
```

14.2.2.14 The PATTERN Keyword

The pattern keyword selects the *first* place within the location source which matches a pattern-matching expression included as the location value. If more than one location matches that expression, there is no error, but the second and later matches are ignored.

Matching is defined to be case-sensitive, i.e. 'abc' is *not* the same as 'ABC'. The pattern is expressed as a regular expression in which the following characters have special meanings, similar to those of many Unix programs (such as *grep*) which handle regular expressions:

. match any single character (including white space characters).

[...] match any character from the set enclosed within the brackets. If, however, the first enclosed character is '^', then match any character *not* from the set enclosed within the brackets. For example, '[^aeiou]' would match any character except a, e, i, o, or u.

**** If the next character is a, d, n, or s, the expression matches any character from a pre-defined group, as shown below; otherwise, the next character is to be taken literally, even if it would otherwise have a special meaning. The special character classes are:

\a any alphabetic character (as defined in the writing system declaration)
\d any digit (0 through 9)
\n any line boundary

\s any white-space character (space, tab, record end, record start)

Note that although \n for newline is provided, its use is discouraged.

***** match zero or more occurrences of the previous regular expression.

+ match one or more occurrences of the preceding regular expression.

? match zero or one occurrences of the preceding regular expression.

^ match the following regular expression only at the beginning of the location source.

$ match the preceding regular expression only at the end of the location source.

| match either the regular expression on the left, or the one on the right.

(...) match the regular expression within the parentheses. (Parentheses are used to control application of the *, ?, +, and | operators, etc.)

For example, the location specification

```
PATTERN (Chapter.8)
```

chooses the first instance of the content string 'Chapter' which is followed by any single character and then the digit 8, within the location source. Various elements which contain that location could be selected by following the `pattern` location term with one or more of other types such as `ancestor` (see above).

It is recommended practice to use structure-oriented location types to specify the destination element as narrowly as possible, and then to specify a pattern only within that element context. If element boundaries are encountered within the location source, however, they are ignored and have no effect on the pattern matching operation.

In formal terms, the location value of the `pattern` keyword is defined thus:

```
regs     ::= '(' regular ')'
          |    regs '(' regular ')'

regular  ::=  character
          |    '.'                              // match any character
          |    '[&circ' || characters || ']'     // match any char not in list
          |    '[' || characters || ']'      // match any char in list
          |    '\a'                      // match any alphabetic
          |    '\d'                      // match any digit 0-9
          |    '\n'                      // match newline (&#RE;&#RS;)
          |    '\s'                      // match any whitespace character
          |    '\\'                      // match backslash (rev. solidus)
          |    '\' || nonspecial         // match nonspecial character
          |    regular || '*'            // match 0-n of 'regular'
          |    regular || '+'            // match 1-n of 'regular'
          |    regular || '?'            // match 0-1 of 'regular'
          |    '&circ' || regular           // match at start of loc source
          |    regular || '$'            // match at end of loc source
          |    regular || regular        // match 1st, then 2d regular exp.
          |    regular || '|' || regular // match either 1st or 2d
          |    '(' || regular || ')'    // use parentheses for grouping

characters ::= /* empty string */
          |    characters character

nonspecial ::= /* any character except a, d, n, or s */
```

14.2.2.15 The TOKEN Keyword

The `token` keyword selects a sequence of one or more *tokens* chosen from within the character content of the location source, where tokens are counted exactly as for the corresponding HyTime `tokenloc` form. The location value must be either a single positive integer, or a pair of positive integers separated by white space, representing the first and the last token numbers to be included in the resulting location. If two integers are specified, the second must not be less than the first. The location source must contain at least as many tokens as are specified in the location value.

This location type should not be used to count across element boundaries. It is recommended practice to use structure-oriented location types to specify the destination element as narrowly as possible, and then to specify a token location only within that element context. If element boundaries are encountered within the location source, they are ignored.

This location type behaves intuitively only for strings containing an alternating sequence of SGML name-characters and white space; this is the type of string found, for example, in SGML attribute values of type IDREFS, such as `a21 z a13`. For compatibility with the HyTime standard, all characters not included in the class of name characters by the current SGML declaration (by default this includes all punctuation other than the hyphen and full stop) are treated as white space characters.

For example, the location specification

```
ID (a27) TOKEN (3 5)
```

chooses the 3rd, 4th, and 5th tokens from the content of the element whose identifier is `a27`. If this element contained the string 'This is _not_ a very good idea', the target selected would be 'not_ a very'.

In formal terms the location value of the `token` and `str` keywords is defined as a range:

```
range    ::=  NUMBER
          |   NUMBER NUMBER
```

14.2.2.16 The STR Keyword

The `str` keyword identifies a sequence of one or more *characters* chosen from within the character content of the location source, where characters are counted exactly as for the HyTime `dataloc` form with `quantum=str`, which has a corresponding meaning and usage. The location value must be either a single positive integer, or a pair of positive integers separated by white space, indicating the first and the last characters to be included in the resulting location. If two integers are specified, the second must not be less than the first. The location source must have at least as many characters as are specified in the larger of the integers.

This location type should not be used to count across element boundaries. The recommended practice is to use structure-oriented location types to specify the destination element, and then to specify a character location only within that element context. If element boundaries are encountered, however, within the location source, they have no effect.

Character offsets in an SGML document must be counted not from the original source file, but from the output of the SGML parser, (the *element structure information set* or *ESIS*). This is because the rules of SGML allow certain characters to be deleted or expanded transparently.

For example, the location specification

```
ID (a27) STRLOC (3 5)
```

chooses the 3rd 4th and 5th characters of the content of the element having identifier a27. If this element contained the string "This turned out to be an even worse idea", the result would be the string 'is ' (i, s and a space).

In multi-byte character sets it is characters which are counted, not bytes. However, in the case of diacritics coded by sequences of bit combinations rather than having separate code points for every combination of letter and diacritic, the diacritics are counted. This means that the following location ladder may retrieve different strings, depending on the system character set in use and on the entity declarations in effect:

```
PATTERN (Wagner's\sG&ouml;tterd&aum;mmerung) STR (10 24)
```

In some character sets, where ö and ä are encoded as single characters, it will select the string 'Götterdämmerung'; in others, where they are encoded with distinct characters for umlaut, a, and o, it will select the string 'Götterdämmeru', truncating the last two letters. If a system-dependent definition is used (containing e.g. a printer escape sequence), the results are even less predictable. For this reason, the `str` keyword must be used with caution and should be avoided where possible.

14.2.2.17 The SPACE Keyword

The `space` location term applies to entities which represent graphical or spatio-temporal data; typically such entities are not encoded in SGML, but in one of many specialized graphical formats. SGML provides standard mechanisms (the NOTATION declaration and related constructs) for specifying what format such an entity uses.

The location value for `space` consists of two or three parenthesized parameter lists. The first contains the name of the co-ordinate space in use. The second and third each consist of any number of signed integers. The numbers in a parameter list represent locations along each dimension of a Cartesian co-ordinate space with all axes orthogonal; the length of the list equals the number of dimensions/axes of the space (usually, but not inevitably, 2, 3, or 4).

If the third parameter list is not specified, the location is the single point in the co-ordinate space specified by the second parameter list. If all three parameter lists are specified, the location is the rectangular prism defined by treating corresponding items of the second and third lists as inclusive bounds along each dimension in turn.

The mapping from co-ordinates to physical or display space, and the meaning and ordering of the axes, are not defined by these guidelines. They should be specified in the TEI header unless they can be determined by definition from the format in which the referenced entity is known to be encoded (for example, many graphics formats can only encode locations in units of pixels, counted in a 3 dimensional left-handed co-ordinate space).

Time may be construed as an axis in addition to any others; when it is, it is TEI recommended practice that it be positioned last. The units used must be defined in the TEI header; it is acceptable in certain media (such as videodiscs) to use frame numbers as a surrogate axis for time.

For example,

```
SPACE (2D) (0 0) (1 1)
```

specifies the location of the unit square tangent to the origin in quadrant 1 of a common graph.

The location value for a `space` location term is a NAME enclosed in parentheses, followed by a point pair:

```
pointpair ::= '(' numbers ')'
          |   '(' numbers ')' '(' numbers ')'

numbers  ::=  signed
          |   numbers signed
```

14.2.2.18 The FOREIGN Keyword

The `foreign` keyword takes any number of parenthesized parameter lists, and is terminated by the end of the attribute value, or by the next non-parenthesized token, whichever comes first.

The meaning of the `foreign` location term is not defined by these Guidelines. It is intended for use in pointing to special kinds of non-SGML, non-coordinate space data. That is, it should be used for making links to data which cannot be specified using the other mechanisms. The meaning of any `foreign` location types must be specified in the TEI header, as a series of paragraphs at the end of the `<encodingDesc>` element defined in section 5.3 ('The Encoding Description') on p. 93.

If more than one such type is used, it is TEI recommended practice that the first parameter list to `foreign` be a name associated with the particular type by documentation in the TEI header.

For example, assume that some program uses a proprietary data format called XFORM, and that the program has supplied an identifier `06286208998` for some piece of data it owns. Then the location specification

```
FOREIGN (XFORM) (06286208998)
```

would be one way of expressing a link to that piece of data.

14.2.2.19 The HYQ Keyword

The HyQ keyword takes a single parenthesized parameter lists, which contains an expression in the HyQ query language defined by the HyTime standard. See documentation on HyTime and HyQ for definitions of HyQ expressions.

14.2.2.20 The DITTO Keyword

The ditto keyword is valid only as the first location term in a ladder, and only within the to attribute of an extended pointer element. It designates the location result of the from attribute on the same element. Thus in the pointer

```
<xptr from='ID (a23) ANCESTOR (1 DIV[0123]) PATTERN (Wagnerian)'
      to='DITTO PATTERN (Liebestod)'>
```

the from attribute designates the first occurrence of the string 'Wagnerian' in the containing the element with an id of a23. The to attribute designates the first occurrence of the string 'Liebestod' which occurs *after* 'Wagnerian', within the same . Without the ditto keyword, it would be necessary to repeat the entire location ladder of the from attribute in the to attribute, which would be error-prone for complex expressions.

14.2.3 Using Extended Pointers

As noted above, when only the from attribute is specified, the <xref> or <xptr> element points at the span indicated by from. When both from and to are specified, the element points at the span running from the beginning of the span indicated by the former to the end of the span indicated by the latter. To point at the second, third, and fourth paragraphs of the second chapter (<div1>) in the body of the current document, therefore, one may specify either of the following:

```
<xptr from='DESCENDANT (1 body) CHILD (2 div1) (2 p)'
      to = 'DESCENDANT (1 body) CHILD (2 div1) (4 p)' >
<!-- or equivalently: -->
<xptr from='DESCENDANT (1 body) CHILD (2 div1) (2 p)'
      to = 'DITTO NEXT (2 p)'>
```

To point to "the <term> occurring in the current <termEntry> with attribute n = 2", only the from attribute would be required:

```
<xptr from='HERE ANCESTOR (1 TERMENTRY) DESCENDANT (1 TERM N 2)'>
```

The following example demonstrates how elements from two different documents may be combined

```
<xptr id=x1 doc=doc1 from='id (d1.1)'>
<xptr id=x2 doc=doc2 from='id (d2.1) tree (2 *)'>

<ptr id=p1 target='x1 x2'>

<link targets='p1 s1 s2' evaluate=all>
```

The first <xptr> indicates the element in doc1 which has identifier d1.1. The second indicates the second subelement of the element in doc2 which has identifier d2.1. These two elements are pointed to as a single item by the <ptr> element and given the identifier p1. This aggregation, finally, is linked with two other elements both in the current document, with identifiers s1 and s2.

An extended pointer, as described above, may specify as its target only a single destination. Where the intended destination of a link is an aggregation or alignment of destinations, possibly in separate documents, an intermediate pointer of some kind must be used, as described in section 14.1.4 ('Intermediate Pointers') on p. 340 elsewhere in this chapter. Like any other element, an <xref> and <xptr> may be given a unique id within the document that contains them. This id value can then be supplied as one of the target values for an intermediate <ptr> or <link> element, to represent aggregation or linkage respectively. The <join> element discussed in section 14.7 ('Aggregation') on p. 369 may also be used.

For example, a modern commentary on an older text must frequently refer to that text, which might well be encoded in a separate SGML document. Some discussions will refer to set of discrete passages in the older text, and will thus require multi-headed pointers. In such a case, the document type declaration must contain a declaration for an SGML entity containing the older text, which might look something like this:

```
<!-- the 1729 Dunciad Variorum -->
<!ENTITY dunciad system 'dunc1729.tei' >
```

In the commentary itself, reference will be made to this external document, using <xptr> and <xref> elements. When the commentary refers to aggregates of discontiguous passages, <xptr> elements are used to point to the individual passage, and a <ref> element may refer to these passages as a group by pointing to the <xptr>s:

```
<xptr id='xL2.5'   doc=dunciad from='ID (L2.5)'  >
<xptr id='xN1.48'  doc=dunciad from='ID (N1.48)' >
<xptr id='xN1.68'  doc=dunciad from='ID (N1.68)' >
<xptr id='xN1.104' doc=dunciad from='ID (N1.104)'>
<xptr id='xN1.106' doc=dunciad from='ID (N1.106)'>
 ...
<p>In <ref target='xL2.5 xN1.48 xN1.68 xN1.104 xN1.106'
evaluate=all>the references to Theobald</ref>, Pope's satire
characteristically ...</p>
```

If the same discontiguous target is to be referred to repeatedly, it may be convenient to give it a single identifier, thus:

```
<xptr id='xL2.5'   doc=dunciad from='ID (L2.5)'  >
<xptr id='xN1.48'  doc=dunciad from='ID (N1.48)' >
<xptr id='xN1.68'  doc=dunciad from='ID (N1.68)' >
<xptr id='xN1.104' doc=dunciad from='ID (N1.104)'>
<xptr id='xN1.106' doc=dunciad from='ID (N1.106)'>
<ptr id='Theobald' target='xL2.5 xN1.48 xN1.68 xN1.104 xN1.106'>
 ...
<p>In <ref target='Theobald' evaluate=all>the references to
Theobald</ref>, Pope's satire characteristically ...</p>
```

A hypertext web might associate passages of the text and notes with the individuals mentioned, the ancient authors imitated, or thematic content, thus:

```
<xptr id='xL2.5'   doc=dunciad from='ID (L2.5)'  >
<xptr id='xN1.48'  doc=dunciad from='ID (N1.48)' >
<xptr id='xN1.68'  doc=dunciad from='ID (N1.68)' >
<xptr id='xN1.104' doc=dunciad from='ID (N1.104)'>
<xptr id='xN1.106' doc=dunciad from='ID (N1.106)'>
<ptr id='Theorefs' target='xL2.5 xN1.48 xN1.68 xN1.104 xN1.106'>

<xptr id='xN2.3'   doc=dunciad from='ID (N2.3)'  >
<xptr id='xN2.46'  doc=dunciad from='ID (N2.46)' >
<xptr id='xN2.54'  doc=dunciad from='ID (N2.54)' >
<xptr id='xN2.66'  doc=dunciad from='ID (N2.66)'>
<xptr id='xN2.78'  doc=dunciad from='ID (N2.78)'>
<xptr id='xN1.104' doc=dunciad from='ID (N1.104)'>
<ptr id='Curlrefs' target='xN2.3 xN2.46 xN2.54 xN2.66 xN2.78 xN1.104'>

 ...
<div><head>Individuals Named in the Text</head>
<list type=gloss>
<label id=curlname>Curll, Edmund</label>
<item  id=curldesc>A bookseller and publisher ...</item>
 ...
<label id=Theoname>Theobald, Lewis.</label>
<item  id=Theodesc>Attorney, active also as editor and reviewer ...
</item>
```

```
 ...
</list>

 ...
<div><head>Ancient Authors Imitated in the Text</head>
<list type=bulleted>
<item id=virgil>Virgil
<item id=homer >Homer
<item id=ovid  >Ovid
 ...
</list>

<ptr id=Curll    target='curlname curldesc'>
<ptr id=Theobald target='theoname theodesc'>
 ...
<linkGrp type=bio>
<link targets='theobald theorefs'>
<link targets='curl     curlrefs'>
 ...
</linkGrp>
<linkGrp type=imitations>
<link targets='virgil   virgrefs'>
<link targets='homer    homerefs'>
 ...
</linkGrp>
```

14.3 Segments and Anchors

In this section, we define two general purposes elements which may be used to mark and categorize both a span of text and a point within one. These elements have several uses, most notably to provide elements which can be given identifiers for use when aligning or linking to parts of a document, as discussed elsewhere in this chapter. They also provide a convenient way of extending the semantics of the TEI markup scheme in a theory-neutral manner.

<seg> contains any arbitrary phrase-level unit of text (including other <seg> elements). Attributes include:

subtype provides a sub-categorization of the segment is marked.

part specifies whether or not the segment is complete. Legal values are:

- ***Y*** the segment is incomplete
- ***N*** either the segment is complete, or no claim is made as to its completeness
- ***I*** the initial part of an incomplete segment
- ***M*** a medial part of an incomplete segment
- ***F*** the final part of an incomplete segment

<anchor> attaches an identifier to a point within a text, whether or not it corresponds with a textual element.

These elements are both members of the class *seg*, and inherit from it the attribute type:

type characterizes the type of segment.
function characterizes the function of the segment.

The <seg> element may be used at the encoder's discretion to mark almost any segment of the text of interest for processing. One use of the element is to mark text features for which no appropriate markup is otherwise defined, i.e. as a simple extension mechanism. Another use is to provide an identifier for some segment which is to be pointed at by some other element, i.e. to provide a target, or a part of a target, for a <ptr> or other similar element.

Several examples of uses for the <seg> element are provided elsewhere in these Guidelines. For example:

- as a means of marking segments significant in a metrical or rhyming analysis (see section 9.4 ('Rhyme and Metrical Analysis') on p. 221)
- as a means of marking typographic lines in drama (see section 10.2 ('The Body of a Performance Text') on p. 235) or title pages (see section 7.5 ('Title Pages') on p. 203)
- as a means of marking prosody- or pause-defined units in transcribed speech (see section 11.3.1 ('Segmentation') on p. 261)
- as a means of marking linguistic or other analyses in a theory-neutral manner (see chapter 15 ('Simple Analytic Mechanisms') on p. 381 passim)

In the following simple example, the <seg> element simply delimits the extent of a stutter, a textual feature for which no element is provided in these Guidelines.

```
<q>Don't say
<q><seg type=stutter>I-I-I</seg>'m afraid,</q>
Melvin, just say
<q>I'm afraid.</q></q>
```

The <seg> element is particularly useful for the mark-up of linguistically significant constituents such as the phrases that may be the output of an automatic parsing system. This example also demonstrates the use of the id attribute to carry an identifier which other parts of a document may use to point to, or align with:

```
<seg type=sentence id=BL0034>
  <seg type=phrase id=BL0034.1>Literate and illiterate speech</seg>
  <seg type=phrase id=BL0034.2>in a language like English</seg>
  <seg type=phrase id=BL0034.3>are plainly different.</seg>
</seg>
```

As the above example shows, <seg> elements may be nested directly within one another, to any degree of analysis considered appropriate. This is taken a little further in the following example, where the type and subtype attributes have been used to further categorise each word of the sentence (the id attributes have been removed to reduce the complexity of the example):

```
<seg type=sentence subtype=declarative>
  <seg type=phrase subtype=noun>
    <seg type=word subtype=adjective>Literate</seg>
    <seg type=word subtype=conjunction>and</seg>
    <seg type=word subtype=adjective>illiterate</seg>
    <seg type=word subtype=noun>speech</seg>
    </seg>
  <seg type=phrase subtype=preposition>
    <seg type=word subtype=preposition>in</seg>
    <seg type=word subtype=article>a</seg>
    <seg type=word subtype=noun>language</seg>
    <seg type=word subtype=preposition>like</seg>
    <seg type=word subtype=noun>English</seg>
    </seg>
  <seg type=phrase subtype=verb>
    <seg type=word subtype=verb>are</seg>
    <seg type=word subtype=adverb>plainly  </seg>
    <seg type=word subtype=adjective>different</seg>
    </seg>
  <seg type=punct>.</seg>
  </seg>
```

(The example values shown are chosen for simplicity of comprehension, rather than verisimilitude). It should also be noted that specialized segment elements are defined in section 15.1 ('Linguistic Segment Categories') on p. 382 to facilitate this particular kind of analysis. These allow for the explicit mark up of units called *s-units*, *clauses*, *phrases*, *words*, *morphemes* and *characters*, which may be felt preferable to the more generic approach typified by use of the <seg> element. Using these, the first phrase above might be encoded simply as

```
<phr type=noun>
  <w type=adjective>Literate</w>
```

```
    <w type=conjunction>and</w>
    <w type=adjective>illiterate</w>
    <w type=noun>speech</w>
  </phr>
```

Note the way in which the type attribute of these specialized elements now carries the value carried by the subtype attribute of the more general <seg> element. For an analysis not using these traditional linguistic categories however, the <seg> element provides a simple but powerful mechanism.

In language corpora and similar material, the <seg> element may be used to provide an end-to-end segmentation as an alternative to the more specific <s> element proposed in chapter 15.1 ('Linguistic Segment Categories') on p. 382 for the mark-up of orthographic sentences, or *s-units*. However, it may be more useful to use the <s> element for this purpose, since this means that the <seg> element can then be used to mark both features within s-units and segments composed of s-units, as in the following example: [7]

```
<seg id=S1S3 type='narrative unit'>
<s id=S1>Sigmund, the
  <seg type=patronymic>son of Volsung</seg>,
   was a king in Frankish country.</s>
<s id=S2>Sinfiotli was the eldest of his sons.</s>
<s id=S3><!-- ... --></s>
</seg>
```

Like other elements, the <seg> tag must be properly enclosed within other elements. Thus, a single <seg> element can be used to group together words in different sentences only if the sentences are not themselves tagged. The first of the following two encodings is legal, but the second is not.

```
Give me <seg type=phrase>a dozen.  Or two or three.</seg>

<!-- Illegal! -->
<s>Give me <seg type=phrase>a dozen.</s>
<s>Or two or three.</s></seg>
```

The part attribute may be used as one simple method of overcoming this restriction:

```
<s>Give me <seg type=phrase part=I>a dozen.</seg></s>
<s><seg part=F>Or two or three.</seg></s>
```

Another solution is to use the <join> element discussed in section 14.7 ('Aggregation') on p. 369; this requires that each of the <seg> elements be given an identifier. For further discussion of this generic encoding problem see also chapter 31 ('Multiple Hierarchies') on p. 633.

The <seg> element has the same content as a paragraph in prose: it can therefore be used to group together consecutive sequences of *inter* class elements, such as lists, quotations, notes, stage directions etc. as well as to contain sequences of phrase-level elements. It cannot however be used to group together sequences of paragraphs or similar text units such as verse lines; for this purpose, the encoder should use intermediate pointers, as described in section 14.1.4 ('Intermediate Pointers') on p. 340 or the methods described in section 14.7 ('Aggregation') on p. 369.

It is particularly important that the encoder provide a clear description of the principles by which a text has been segmented, and the way in which that segmentation is represented. This should include a description of the method used and the significance of any categorization codes. The description should be provided as a series of paragraphs within the <segmentation> element of the encoding description in the TEI header, as described in section 5.3.3 ('The Editorial Practices Declaration') on p. 96.

The remainder of this chapter contains a number of examples of the use of the <seg> element simply to provide an element to which an identifier may be attached, for example so that another segment may be linked or related to it in some way. We conclude this section by introducing the <anchor> element which serves an identical purpose, but has no content. It may be thought of

[7] See section 15.3 ('Spans and Interpretations') on p. 387, where the text from which this fragment is taken is analyzed.

as an empty <seg>, or as an artifice enabling an identifier to be attached to any position in a text.

Like the <milestone> element discussed in section 6.9 ('Reference Systems') on p. 155, the <anchor> element is useful where multiple views of a document are to be combined, for example, when a logical view based on paragraphs or verse lines is to be mapped on to a physical view based on manuscript lines. It differs from the milestone and related elements in that the <anchor> element should not be used to mark the start or end of an arbitrary zone within a text, but only to mark arbitrary points used for alignment.

For example, suppose that we wish to mark the end of the fifth word following each occurrence of some term in a particular text, perhaps to assist with some collocational analysis. This can most easily be done with the help of the <anchor> tag, as follows:

```
<!-- ... -->
English language.  Except for not very<anchor id=eng1 >
<!-- ... -->
English at all at the time<anchor id=eng2 >
<!-- ... -->
English was still full of flaws<anchor id=eng3 >
<!-- ... -->
English.  This was revised by young<anchor id=eng4>
```

In the next section we discuss ways in which these <anchor> points can be used to represent an alignment, for example such as one might get in a keyword-in-context concordance.

These elements are formally defined as follows:

```
<!-- 14.3:  Segments and Anchors                              -->
<!ELEMENT seg           - -  (%paraContent)                    >
<!ATTLIST seg                %a.global;
                             %a.seg;
          subtype            CDATA               #IMPLIED
          part               (Y | N | I | M | F) N              >
<!ELEMENT anchor        - O  EMPTY                              >
<!ATTLIST anchor
          n                  CDATA               #IMPLIED
          lang               IDREF               %INHERITED
          rend               CDATA               #IMPLIED
                             %a.seg;
          id                 ID                  #REQUIRED      >
<!-- This fragment is used in sec. 14                          -->
```

14.4 Correspondence and Alignment

In this section we introduce the notions of *correspondence*, expressed by the corresp attribute, and of *alignment*, which is a special kind of correspondence involving an ordered set of correspondences. Both cases may be represented using the <link> and <linkGrp> elements introduced in section 14.1 ('Pointers') on p. 333. We also discuss the special case of alignment in time or *synchronization*, for which special purpose elements are proposed in section 14.5 ('Synchronization') on p. 364.

14.4.1 Correspondence

A common problem in text analysis is to determine correspondences between two or more parts of a single document, or between places in different documents. Provided that SGML elements are available to represent the parts or places to be linked, then the global linking attribute corresp may be used to encode such correspondence, once it has been identified.

corresp points to elements that correspond to the current element in some way.

This is one of the attributes made available by the mechanism described in the introduction to this chapter (14 ('Linking, Segmentation, and Alignment') on p. 331). Correspondence can also

be expressed by means of the <link> element introduced in section 14.1 ('Pointers') on p. 333.

Where the correspondence is between *spans*, the <seg> element should be used, if no other element is available. Where the correspondence is between *points*, the <anchor> element should be used, if no other element is available.

The use of the corresp attribute with spans of content is illustrated by the following example:

```
<title id=shirley>Shirley</title>, which made
its Friday night debut only a month ago, was
not listed on <name id=nbc>NBC</name>'s new schedule,
although <seg id=network corresp=nbc>the network</seg>
says <seg id=show corresp=shirley>the show</seg>
still is being considered.
```

Here the anaphoric phrases 'the network' and 'the show' have been associated directly with the elements to which they refer by means of corresp attributes. This mechanism is simple to apply, but has the drawback that it is not possible to specify more exactly what kind of correspondence is intended. Where this attribute is used, therefore, encoders are encouraged to specify their intent in the associated encoding declarations in the TEI Header.

Essentially, what the corresp attribute does is to specify that the element that has the attribute and the element(s) the attribute points to are doubly linked. [8] Therefore, we can also use the <link> and <linkGrp> elements defined in section 14.1 ('Pointers') on p. 333 to indicate correspondence among elements. Moreover, the use of these elements provides a convenient place to indicate what kind of correspondence is intended as in the following retagging of the preceding example.

```
<title id=shirley>Shirley</title>, which made
its Friday night debut only a month ago, was not
listed on <name id=nbc>NBC</name>'s new schedule,
although <seg id=network>the network</seg> says
<seg id=show>the show</seg> still is being considered.
<!-- ... -->
<linkGrp type='anaphoric link'
        targFunc='antecedent anaphor' targOrder=Y>
  <link targType='title seg' targets='shirley show'>
  <link targType='name seg' targets='nbc network'>
</linkGrp>
```

In the following example, we use exactly the same mechanism to express a correspondence amongst the anchors introduced following the fifth word after 'English' in a text:

```
English language.  Except for not very<anchor id=eng1>
<!-- ... -->
English at all at the time<anchor id=eng2>
<!-- ... -->
English was still full of flaws<anchor id=eng3>
<!-- ... -->
English.  This was revised by young<anchor id=eng4>

<linkGrp type='five-word collocates'
         targType='anchor anchor'
         targOrder=N>
  <link type='collocates of ENGLISH' targets='eng1 eng2 eng3 eng4'>
  <!-- ... -->
</linkGrp>
```

[8] The corresp attribute is thus distinct from the target attribute in that it is understood to create a double, rather than a single, link. It is also distinct from the targets attribute in that the latter lists all the identifiers of the elements that are doubly linked, whereas the corresp doubly links the element that bears the attribute with the element(s) that make up the value of the attribute.

14.4.2 Alignment of Parallel Texts

One very important application area for the alignment of parallel texts is multilingual corpora. Consider, for example, the need to align "translation pairs" of sentences drawn from a corpus such as the Canadian Hansard, in which each sentence is given in both English and French.

Concerning this problem, Gale and Church write:[9]

"Most English sentences match exactly one French sentence, but it is possible for an English sentence to match two or more French sentences. The first two English sentences [in the example below] illustrate a particularly hard case where two English sentences align to two French sentences. No smaller alignments are possible because the clause "...sales...were higher..." in the first English sentence corresponds to (part of) the second French sentence. The next two alignments ... illustrate the more typical case where one English sentence aligns with exactly one French sentence. The final alignment matches two English sentences to a single French sentence. These alignments [which were produced by a computer program] agreed with the results produced by a human judge."

The alignment produced by Gale and Church's program can be expressed in four different ways. The encoder must first decide whether to represent the alignment in terms of points within each text (using the <anchor> element) or in terms of whole stretches of text, using the <seg> element. To some extent the choice will depend on the process by which the software works out where alignment occurs, and the intention of the encoder. Secondly, the encoder may elect to represent the actual encoding using either corresp attributes attached to the individual <anchor> or <seg> elements, or using a free standing <linkGrp> element.

We present first a solution using <anchor> elements bearing only corresp attributes:

```
<div1 id=E lang=EN>
<p><anchor id=EA1 corresp=FA1>According to our survey, 1988
sales of mineral water and soft drinks were much higher than
in 1987, reflecting the growing popularity of these
products.  Cola drink manufacturers in particular achieved
above-average growth rates.  <anchor id=EA2 corresp=FA2>The
higher turnover was largely due to an increase in the sales
volume.  <anchor id=EA3 corresp=FA3>Employment and
investment levels also climbed.  <anchor id=EA4
corresp=FA4>Following a two-year transitional period, the
new Foodstuffs Ordinance for Mineral Water came into effect
on April 1, 1988.  Specifically, it contains more stringent
requirements regarding quality consistency and purity
guarantees.  </div1>
<!-- ... -->
<div1 id=F lang=FR>
<p><anchor id=FA1 corresp=EA1>Quant aux eaux min&eacute;rales
et aux limonades, elles rencontrent toujours plus
d'adeptes.  En effet, notre sondage fait ressortir des
ventes nettement sup&eacute;rieures &agrave; celles de 1987,
pour les boissons &agrave; base de cola notamment.  <anchor
id=FA2 corresp=EA2>La progression des chiffres d'affaires
r&eacute;sulte en grande partie de l'accroissement du volume
des ventes.  <anchor id=FA3 corresp=EA3>L'emploi et les
investissements ont &eacute;galement augment&eacute;.
<anchor id=FA4 corresp=EA4>La nouvelle ordonnance
f&eacute;d&eacute;rale sur les denr&eacute;es alimentaires
concernant entre autres les eaux min&eacute;rales,
entr&eacute;e en vigueur le 1er avril 1988 apr&egrave;s une
p&eacute;riode transitoire de deux ans, exige surtout une
plus grande constance dans la qualit&eacute; et une garantie
de la puret&eacute;.  </div1>
```

[9] See William A. Gale and Kenneth W. Church, *Program for aligning sentences in bilingual corpora*, *Computational Linguistics* 19 (1993): 75-102, from which the example in the text is taken.

There is no requirement that the corresp attribute be specified in both English and French texts, since (as noted above) this attribute is defined as representing a mutual association. However, it may simplify processing to do so, and also avoids giving the impression that the English is translating the French, or vice versa. More seriously, this encoding does not make explicit the fact that it is in fact the entire stretch of text between the anchors which is being aligned, not simply the points themselves. If for example one text contained material omitted from the other, this approach would not be appropriate.

We now present the same passage using the alternative <linkGrp> mechanism and marking explicitly the segments which have been aligned:

```
<div1 id=E lang=EN>
<p><seg id=E1>According to our survey, 1988 sales of mineral
water and soft drinks were much higher than in 1987,
reflecting the growing popularity of these products.  Cola
drink manufacturers in particular achieved above-average
growth rates.</seg>
<seg id=E2>The higher turnover was largely due to an
increase in the sales volume.</seg>
<seg id=E3>Employment and investment levels also
climbed.</seg>
<seg id=E4>Following a two-year transitional period, the new
Foodstuffs Ordinance for Mineral Water came into effect on
April 1, 1988.  Specifically, it contains more stringent
requirements regarding quality consistency and purity
guarantees.</seg>
</div1>
<!-- ... -->
<div1 id=F lang=FR>
<p><seg id=F1>Quant aux eaux min&eacute;rales et aux limonades,
elles rencontrent toujours plus d'adeptes.  En effet, notre
sondage fait ressortir des ventes nettement
sup&eacute;rieures &agrave; celles de 1987, pour les
boissons &agrave; base de cola notamment.</seg>
<seg id=F2>La progression des chiffres d'affaires
r&eacute;sulte en grande partie de l'accroissement du volume
des ventes.</seg>
<seg id=F3>L'emploi et les investissements ont
&eacute;galement augment&eacute;.</seg>
<seg id=F4>La nouvelle ordonnance f&eacute;d&eacute;rale sur
les denr&eacute;es alimentaires concernant entre autres les
eaux min&eacute;rales, entr&eacute;e en vigueur le 1er avril
1988 apr&egrave;s une p&eacute;riode transitoire de deux
ans, exige surtout une plus grande constance dans la
qualit&eacute; et une garantie de la puret&eacute;.</seg>
</div1>
<!-- ... -->
<linkGrp type=alignment
         domains='E F'
  <link targets='E1 F1'>
  <link targets='E2 F2'>
  <link targets='E3 F3'>
  <link targets='E4 F4'>
</linkGrp>
```

Note that use of the <seg> element allows us to mark up the orthographic sentences in both languages independently of the alignment: the first translation pair in this example might be marked up as follows:

```
<div1 id=E lang=EN>
<seg id=E1>
  <s>According to our survey, 1988 sales of mineral water and soft
  drinks were much higher than in 1987, reflecting the growing popularity
```

```
   of these products.</s>
  <s>Cola drink manufacturers in particular achieved above-average
   growth rates.</s>
</seg>

<!-- ... -->

<div1 id=F lang=FR>
<seg id=F1>
  <s id=fs1>Quant aux eaux min&eacute;rales et aux limonades, elles
   rencontrent toujours plus d'adeptes.</s>
  <s id=fs2>En effet, notre sondage fait ressortir des ventes nettement
   sup&eacute;rieures &agrave; celles de 1987, pour les boissons &agrave;
   base de cola notamment. </s>
</seg>
```

14.4.3 A Three-way Alignment

The preceding encoding of the alignment of parallel passages from two texts requires that those texts and the alignment all be part of the same SGML document. If the texts are in separate documents, then additional <xptr> elements must be supplied, as discussed in section 14.2 ('Extended Pointers') on p. 340. These external pointers may appear anywhere within the document, but if they are created solely for use in encoding links, they may for convenience be grouped within the <linkGrp> (or other grouping element that uses them for linking).

To demonstrate this facility, we consider how we might encode the alignments in an extract from Comenius' *Orbis Sensualium Pictus*. The figure shows the page from the *Orbis pictus* of Comenius which is discussed in the text. Each topic covered in this work has three parts: a picture, a prose text in Latin describing the topic, and a carefully-aligned translation of the Latin into English, German or some other vernacular. Key terms in the two texts are typographically distinct, and are linked to the picture by numbers, which appear in the two texts and within the picture as well.[10]

First, we present the text portions. The English and Latin portions have been encoded as distinct elements. Identifiers have been attached to each typographic line, but no other encoding added, to simplify the example.

```
<!-- English text -->

<div id=E98 lang=EN><head>The Study</head>
<seg id=E9801>The Study</seg>
<seg id=E9802>is a place</seg>
<seg id=E9803>where a Student,</seg>
<seg id=E9804>a part from men,</seg>
<seg id=E9805>sitteth alone,</seg>
<seg id=E9806>addicted to his Studies,</seg>
<seg id=E9807>whilst he readeth</seg>
<seg id=E9808>Books,</seg>
<!-- ... -->
</div>

<!-- Latin text -->

<div id=L98 lang=LA><head>Mus&eacute;um</head>
<seg id=L9801>Museum</seg>
<seg id=L9802>est locus</seg>
<seg id=L9803>ubi Studiosus,</seg>
```

[10] Our example uses the English translation of Charles Hoole (1659), and is taken from John E. Sadler, ed., *John Amos Comenius Orbis Pictus: a facsimile of the first English edition of 1659* (Oxford: Oxford University Press, 1968) (The Juvenile Library).

```
<seg id=L9804>secretus ab hominibus,</seg>
<seg id=L9805>solus sedet,</seg>
<seg id=L9806>Studiis deditus,</seg>
<seg id=L9807>dum lectitat</seg>
<seg id=L9808>Libros,</seg>
<!-- ... -->
</div>
```

Next we assume that we have stored a digitized image of the picture itself in some external entity we will call *com98* (for further discussion of the handling of external images and graphics, see section 22.3 ('Specific Elements for Graphic Images') on p. 530). We further assume that we can address portions of this image as a two-dimensional co-ordinate space. The SPACE location method of the <xptr> element (discussed in section 6.6 ('Simple Links and Cross References') on p. 147 above) can now be used to point to the whole picture and to two portions of it, one containing the picture of a student and the other of a book, as follows:

```
<xptr n='1' id=p981 doc=com98>
<xptr n='2' id=p982 doc=com98 from='space (2d) (75 5) (133 75)'>
<xptr n='3' id=p983 doc=com98 from='space (2d) (55 42) (90 60)'>
```

Note that each external pointer has its own unique identifier, in addition to the n attribute, which last holds the visible label (or "explainer") used for this image portion in the original.

As printed, the text exhibits three kinds of alignment.

1. The English and Latin portions are printed in two parallel columns, with corresponding phrases, (represented above by <seg> elements), more or less next to each other.
2. Particular words or phrases are marked as terms in the two languages by a change of rendition: the English text, which otherwise uses black letter type throughout, has the words 'The Study', 'a Student', 'Studies', and 'Books' in a roman font; in the Latin text, which is printed in roman, the corresponding words ('Museum', 'Studiosus', 'Studiis', and 'Libros') are all in italic.
3. Numbered labels appear within the text portions, linking keywords to each other and to sections of the picture. These labels, which have been left out of the above encoding, are attached to the first third and last segment in each language quoted below, and also appear (rather indistinctly) within the picture itself. If it is desired to transcribe them in the text, they might be encoded using as <ref> elements, <anchor> elements, or <xptr>s to the picture; the number itself would be transcribed as the value of the n attribute (or as the content of the <ref>).

The first kind of alignment might be represented by using the corresp attribute on the <seg> element. The second kind might be represented by using the <gloss> and <term> mechanism described in section 6.3.4 ('Terms, Glosses, and Cited Words') on p. 130. The third kind of alignment might be represented using pointers embedded within the texts, although this would involve some duplication. We choose however to use the <link> element, since this provides an efficient way of representing the three-way alignment between English, Latin and picture without redundancy.

```
<linkGrp type=alignment>
  <link targets='E9801 L9801 p981'>
  <link targets='E9802 L9802     '>
  <link targets='E9803 L9803 p982'>
  <link targets='E9804 L9804     '>
  <link targets='E9805 L9805     '>
  <link targets='E9806 L9806     '>
  <link targets='E9807 L9807     '>
  <link targets='E9808 L9808 p983'>
</linkGrp>
```

This map, of course, only aligns whole segments and image portions, since these are the only parts of our encoding which bear identifiers and can therefore be pointed to. To add to it the alignment between the typographically distinct words mentioned above, new elements must be defined, either within the text itself or externally by using the extended pointer mechanism. Encoding these word pairs as <term> and <gloss>, although intuitively obvious, requires a

non-trivial decision as to whether the Latin text is glossing the English, or vice-versa. Tagging all the marked words as <term> avoids the difficult decision, but might be thought by some encoders to convey the wrong information about the words in question. Simply tagging them as additional embedded <seg> elements with identifiers that can be aligned like the others is also a possibility. All of these require the addition of further markup to the text. This may pose no problems, or it may be infeasible (e.g. if the text is held on a read-only medium). If it is not feasible to add more markup to the original text, the extended pointer mechanism is likely to be the best choice. For example, to indicate that the words 'Studies' and 'Studiis' correspond, two external pointers might be defined and aligned as follows:

```
<xptr id=xt981  from='id (s E9806) token (4)'>
<xptr id=xt982  from='id (s L9806) token (1)'>
<link target='xt981 xt982'>
```

14.5 Synchronization

In the previous section we discussed two particular kinds of alignment: alignment of parallel texts in different languages; and alignment of texts and portions of an image. In this section we address another specialized form of alignment: synchronization. The need to mark the relative positions of text components with respect to time arises most naturally and frequently in transcribed spoken texts, but it may arise in any text in which quoted speech occurs, or events are described within a time frame. The methods described here are also generalizable for other kinds of alignment (for example, alignment of text elements with respect to space), and may thus be regarded as providing a simplified version of the HyTime system of *finite space co-ordinates*.

14.5.1 Aligning Synchronous Events

To mark synchronous elements, the synch attribute, which is one of the linking attributes that are available for all text elements, may be used.

synch points to elements that are synchronous with the current element.

Alternatively, the <link> and <linkGrp> elements may be used to make explicit the fact that the synchronous elements are aligned.

To illustrate the use of these mechanisms for marking synchrony, consider the following representation of a spoken text:

```
B:   The first time in twenty five years, we've cooked Christmas
     (unclear) for a blooming great load of people.
A: So you're [1] (unclear) [2]
B: [1] It will be [2] nice in a way, but, [3] be strange. [4]
A: [3] Yeah [4], yeah, cos it, it's [5] the [6]
B: [5] not [6]
```

This representation uses numbers in brackets to mark the points at which speakers overlap each other. For example, the '[1]' in A's first speech is to be understood as coinciding with the '[1]' in B's second speech.[11]

To encode this we use the base tag set for spoken texts, described in chapter 11 ('Transcriptions of Speech') on p. 249, together with the additional tag set described in the present chapter. First, we transcribe this text, marking the synchronous points with <anchor> elements, and providing a synch attribute on one of each of the pairs of synchronous anchors. As noted in the example given above (section 14.4.2 ('Alignment of Parallel Texts') on p. 360), correspondence, and hence synchrony, is a symmetric relation; therefore the attribute need only be specified on one of the pairs of synchronous anchors.

```
<div id=d1>
<!-- ... -->
```

[11] This sample is taken from a conversation collected and transcribed for the British National Corpus.

```
<u id=u3a who=A>So you're <anchor id=t1a synch=t1b><unclear>
          <anchor id=t2a synch=t2b></u>
<u id=u3b who=B> <anchor id=t1b>It will be <anchor id=t2b>
          nice in a way, but, <anchor id=t3b>
          be strange.<anchor id=t4b></u>
<u id=u4a who=A>
          <anchor id=t3a synch=t3b>Yeah
          <anchor id=t4a synch=t4b>, yeah, cos it, its
          <anchor id=t5a synch=t5b>the
          <anchor id=t6a synch=t6b></u>
<u id=u4b who=B>
          <anchor id=t5b>not<anchor id=t6b>
</u>
```

Next, we encode the same example using <link> and <linkGrp> elements to make the temporal alignment explicit; the id attributes are provided for the <link> and <linkGrp> elements for a reason that is given in the next section, 14.5.2 ('Placing Synchronous Events in Time') on p. 366.

```
<div id=d1>
<u id=u2b who=B>The first time in twenty five years,
  we've cooked Christmas <unclear> for a blooming great
  load of people.</u>
<u id=u3a who=A>So you're
  <anchor id=t1a><unclear><anchor id=t2a></u>
<u id=u3b who=B>
  <anchor id=t1b>It will be <anchor id=t2b>nice in a way,
  but, <anchor id=t3b>be strange <anchor id=t4b></u>
<u id=u4a who=A>
  <anchor id=t3a>Yeah<anchor id=t4a>, yeah, cos it, it's
<anchor id=t5a>the<anchor id=t6a></u>
<u id=u4b who=B><anchor id=t5b>not<anchor id=t6b></u>
</div>
<!-- ... -->
<linkGrp id=lg1 type='synchronous alignment'
         domains='d1 d1'
         targFunc='speaker.A speaker.B'>
  <link id=l1 targets='t1a t1b'>
  <link id=l2 targets='t2a t2b'>
  <link id=l3 targets='t3a t3b'>
  <link id=l4 targets='t4a t4b'>
  <link id=l5 targets='t5a t5b'>
  <link id=l6 targets='t6a t6b'>
</linkGrp>
```

As with other forms of alignment, synchronization may be expressed between stretches of speech as well as between points. When complete utterances are synchronous, for example, if one person says 'What?' and another 'No!' at the same time, that can be represented without <anchor> elements as follows.

```
<u id=u1 who=A synch=u2>What?</u>
<u id=u2 who=B>No!</u>
```

A simple way of expressing *overlap* (where one speaker starts speaking before another has finished) is thus to use the <seg> element to encode the overlapping portions of speech. For example,

```
<u who=A> So you're <unclear synch=B1></u>
<u who=B><seg id=B1> It will be </seg> nice in a way, but,
         <seg synch=A3> be strange. </seg></u>
<u who=A><seg id=A3> Yeah </seg>, yeah, cos it,
         its <seg synch=B2> the </seg></u>
<u who=B id=B2> not </u>
```

Note in this encoding how synchronization has been effected between an empty <unclear> element and a <seg>, and between an entire <u> element and another <seg>, using the synch attribute. Alternatively, a <linkGrp> could be used in the same way as above.

14.5.2 Placing Synchronous Events in Time

A synchronous alignment specifies which points in a spoken text occur at the same time, and the order in which they occur, but does not say at what time those points actually occur. If that information is available to the encoder it can be represented by means of the <when> and <timeline> elements, whose description and attributes are the following:

<when> indicates a point in time either relative to other elements in the same <timeline> tag, or absolutely. Attributes include:

- **absolute** supplies an absolute value for the time.
- **interval** specifies the numeric portion of a time interval
- **unit** specifies the unit of time corresponding to the interval value.
- **since** identifies the reference point for determining the time of the current <when> element, which is obtained by adding the interval to the time of the reference point.
- **id** supplies an identifier, unique to the document, for each <when> element.

<timeline> provides a set of ordered points in time which can be linked to elements of a spoken text to create a temporal alignment of that text. Attributes include:

- **origin** designates the origin of the timeline, i.e. the time at which it begins.
- **interval** specifies the numeric portion of a time interval
- **unit** specifies the unit of time corresponding to the interval value of the timeline or of its constituent points in time.

Each <when> element indicates a point in time, either directly by means of the absolute attribute, whose value is a string which specifies a particular time, or indirectly by means of the since attribute, which points to another <when>. If the since is used, then the interval and unit attributes should also be used to indicate the amount of time that has elapsed since the time specified by the element pointed to by the since attribute; the value -1 can be given to indicate that the interval is unknown.

If the <when> elements are uniformly spaced in time, then the interval and unit values need be given once in the <timeline>, and not repeated in any of the <when> elements. If the intervals vary, but the units are all the same, then the unit attribute alone can be given in the <timeline> element, and the interval attribute given in the <when> element.

The origin attribute in the <timeline> element points to a <when> element which specifies the reference or origin for the timings within the <timeline>; this must, of course, specify its position in time absolutely.

The following <timeline> might be used to accompany the marked up conversation shown in the preceding section:

```
<timeline id=tl1 origin=w0 unit=centisecond>
  <when id=w0 absolute='sometime Monday morning before Christmas'
  <when id=w1 since=w0 interval=-1>
  <when id=w2 since=w1 interval=10>
  <when id=w3 since=w2 interval=20>
  <when id=w4 since=w3 interval=15>
  <when id=w5 since=w4 interval=25>
  <when id=w6 since=w5 interval=10>
  </timeline>
```

The information in this <timeline> could now be linked to the information in the <linkGrp> which provides the temporal alignment (synchronization) for the text, as follows:

```
<linkGrp type='temporal specification'
         targType='link when'
         targFunc='synch.points when'
         domains='lg1 tl1'>
  <link targets='l1 w1'>
```

```
  <link targets='l2 w2'>
  <link targets='l3 w3'>
  <link targets='l4 w4'>
  <link targets='l5 w5'>
  <link targets='l6 w6'>
  </linkGrp>
```

To avoid the need for two distinct link groups (one marking the synchronization of anchors with each other, and the other marking their alignment with points on the time line) it would be better to link the <when> elements with the synchronous points directly:

```
<linkGrp type='temporal specification'
         targType='anchor anchor when'
         targFunc='speaker.A speaker.B when'
         domains='d1 d1 tl1'>
  <link targets='t1a t1b w1'>
  <link targets='t2a t2b w2'>
  <link targets='t3a t3b w3'>
  <link targets='t4a t4b w4'>
  <link targets='t5a t5b w5'>
  <link targets='t6a t6b w6'>
  </linkGrp>
```

Finally, suppose that a digitized audio recording is also available. The extended pointer syntax described in section 14.2 ('Extended Pointers') on p. 340 could be used to address positions on or portions of this recording directly. Assuming that <xptr> elements with identifiers X1, X2, etc., have been defined to do this, these identifiers could also be included as a fourth component in each of the above <link> elements, thus providing a synchronized audio track to complement the transcribed text.

For further discussion of this and related aspects of encoding transcribed speech, refer to chapter 11 ('Transcriptions of Speech') on p. 249.

The <when> and <timeline> elements are defined as follows:

```
<!-- 14.5.2:  Temporal specification                          -->
<!ELEMENT when          - O  EMPTY                             >
<!ATTLIST when
          n                  CDATA              #IMPLIED
          lang               IDREF              %INHERITED
          rend               CDATA              #IMPLIED
          absolute           CDATA              #IMPLIED
          unit               NMTOKEN            %INHERITED;
          interval           NUTOKEN            %INHERITED;
          since              IDREF              #IMPLIED
          id                 ID                 #REQUIRED      >
<!ELEMENT timeline      - -  ((when)+)                         >
<!ATTLIST timeline           %a.global;
          origin             IDREF              #REQUIRED
          unit               NMTOKEN            #IMPLIED
          interval           NUTOKEN            #IMPLIED       >
<!-- This fragment is used in sec. 14                         -->
```

14.6 Identical Elements and Virtual Copies

This section introduces the notion of a *virtual element*, that is, an element which is not explicitly present in a text, but the presence of which an application can infer from the encoding supplied. In this section, we are concerned with virtual elements made by simply cloning existing elements. In the next section (14.7 ('Aggregation') on p. 369), we discuss virtual elements made by aggregating existing elements.

It is useful to be able to represent the fact that one element of text is identical to others, for analytical purposes, or (especially if the elements have lengthy content) to obviate the need to

repeat the content. For example, consider the repetition of the <date> element in the following material:

```
<p>In small clumsy letters he wrote:
<q rend='centered italic' id=d1>
<date id=D840404>April 4th, 1984</date>.
</q></p>
<p>He sat back. A sense of complete helplessness had
descended upon him.
<!-- ... -->
<p>His small but childish handwriting straggled up
and down the page, shedding first its capital letters
and finally even its full stops:
<!-- ... -->
<q rend=italic id=d2>
<date>April 4th, 1984</date>. Last night to the flicks.
<!-- ... -->
</q></p>
```

Suppose now that we wish to encode the fact that the second <date> element above has identical content to the first. The sameAs attribute is provided for this purpose. Using it, we can recode the last line of the above example as follows:

```
<!-- ... -->
<date sameAs=D840404>April 4th, 1984</date>. Last night to the flicks.
<!-- ... -->
```

The sameAs attribute may be used to document the fact that two elements have identical content. It may be regarded as a special kind of link. It should only be attached to an element with identical content to that it indicates, or to one the content of which clearly designates it as a repetition, such as the word 'repeat' or 'bis' in the representation of the chorus of a song, the second time it is to be sung. The relation specified by the sameAs attribute is symmetric: if a chorus is repeated three times and each repetition bears a sameAs attribute indicating the first occurrence of the element concerned, it is implied that each chorus is identical, and there is no need for the first occurrence to specify any of its copies.

The copyOf attribute is used in a similar way to indicate that the content of the element bearing it is identical to that of another. The difference is that the content is not itself repeated. The effect of this attribute is thus to create a *virtual copy* of the element indicated. Using this attribute, the repeated date in the first example above could be recoded as follows:

```
<!-- ... -->
<date copyOf=D840404></date>
```

An application program should replace whatever is the actual content of an element bearing a copyOf attribute with the content of the element specified by it. If the content of the element specified includes other elements, these will become embedded within the element bearing the attribute. Care must be taken to ensure that the document is a legal SGML document both before and after this embedding takes place. If, for example, the element bearing a copyOf attribute requires a mandatory sub-component, then this component must be present (though possibly empty), even though it will be replaced by the content of the targetted element.

The following example demonstrates how the copyOf attribute may be used in conjunction with the <seg> element to highlight the differences between almost identical repetitions:

```
<sp who=Mikado>
<l>My <seg id=L1s>object all sublime</seg></l>
<l>I shall <seg id=L2s>achieve in time</seg>&dash;</l>
<l id=L3>To let
  <seg id=L3s>the punishment fit the crime</seg>,</l>
<l id=L4><seg copyOf=L3s></seg>;</l>
<l id=L5>And make each pris'ner pent</l>
<l id=L6>Unwillingly represent</l>
<l id=L7>A source
  <seg id=L7s>of innocent merriment</seg>,</l>
```

```
<l id=L8><seg copyOf=L7s></seg>!</l>
</sp>
<sp who=chorus>
<l>His <seg copyOf=L1s></seg></l>
<l>He will <seg copyOf=L2s></seg></l>
<l copyOf=L3></l>
<l copyOf=L4></l>
<l copyOf=L5></l>
<l copyOf=L6></l>
<l copyOf=L7></l>
<l copyOf=L8></l>
</sp></l>
```

For further examples of the use of this attribute, see chapters 21 ('Graphs, Networks, and Trees') on p. 505 and 16 ('Feature Structures') on p. 397, where it is used to reduce the complexity of formal analytic representations of structure.

14.7 Aggregation

Because of the strict hierarchical organization of an SGML document, or for other reasons, it may not always be possible or desirable to include all the parts of a possibly fragmented text segment within a single element. In section 14.1.4 ('Intermediate Pointers') on p. 340 we introduced the notion of an intermediate pointer as a way of pointing to discontinuous segments of this kind. In this section we first describe another way of linking the parts of a discontinuous whole, using a set of linking attributes, which are made available for any tag by following the procedure described at the beginning of this chapter. We then describe how the <link> element may be used to aggregate such segments, and finally introduce the <join> element, which is a special-purpose linking element specifically for representing the aggregation of parts, and the <joinGrp> for grouping <join> tags.

The linking attributes for aggregation are next and prev; each of these attributes has a single identifier as its value:

next points to the next element of a virtual aggregate of which the current element is part.
prev points to the previous element of a virtual aggregate of which the current element is part.

The <join> element is also a member of the class of *pointer* elements, and so may carry any of the attributes of that class; for the list, see section 14.1 ('Pointers') on p. 333.

Here is the material on which we base our first illustration of the use of these mechanisms. Our problem is to represent the S-units identified below as qs3 and qs4 as a single (but discontinuous) whole:

```
<q who=waitress>
 <s id=qs2>Monsieur Paul, after he has taken equal
  parts of goose breast and the finest pork, and
  broken a certain number of egg yolks into them,
  and ground them <emph>very</emph>, very fine,
  cooks all with seasoning for some three hours.</s>
<s id=qs3><emph>But</emph>,</s>
</q>
<s id=ps2>she pushed her face nearer, and looked with
 ferocious gloating at the p&acirc;t&eacute;
 inside me, her eyes like X rays,</s>
<q who=waitress>
  <s id=qs4>he never stops stirring it!</s>
  <s id=qs5>Figure to yourself the work of it &mdash;
  <s id=qs6>stir, stir, never stopping!</s>
</q>
```

Using the prev and next attributes, we can link the s-units with identifiers s1 and s2, either singly or doubly as follows:

```
<s id=qs3 next=qs4><emph>But</emph>,</s>
<!-- intervening material -->
<s id=qs4>he never stops stirring it!</s>

<s id=qs3><emph>But</emph>,</s>
<!-- intervening material -->
<s id=qs4 prev=qs3>he never stops stirring it!</s>

<s id=qs3 next=qs4><emph>But</emph>,</s>
<!-- intervening material -->
<s id=qs4 prev=qs3>he never stops stirring it!</s>
```

Double linking of the two S-units, as illustrated by the last of these encodings, is equivalent to specifying a <link> tag:

```
<link type=join targType='s s' targOrder=Y targets='qs3 qs4'>
```

Such a <link> element must carry type=join attribute value to specify that the link is to be understood as joining its targets into a single aggregate.

The <join> element is equivalent to a <link> element of type `join`; unlike a link, the default value for the targOrder attribute which this element also inherits from the pointer class is Y. Also unlike the <link> element, the <join> element can additionally specify information about the virtual element which it represents, by means of its result attribute. And finally, unlike the <link> element, the position of a <join> element within a text is significant: it must be supplied at a position where the element indicated by its result attribute would be contextually legal.

<join> identifies a possibly fragmented segment of text, by pointing at the possibly discontiguous elements which compose it. Attributes include:

- **result** specifies the name of an element which this aggregation may be understood to represent.
- **targets** specifies the SGML identifiers of the elements or passages to be joined into a virtual element.
- **targOrder** specifies whether or not the order in which components of the join are listed on the targets attribute is significant. Legal values are:
  - ***Y*** Yes: the order should be followed when combining the targeted elements.
  - ***N*** No: the order has no significance when combining the targeted elements.
  - ***U*** Unspecified: the order may or may not be significant.

<joinGrp> groups a collection of <join> elements and possibly pointers. Attributes include:

- **result** describes the result of the joins gathered in this collection.

To conclude the above example, we now use a <join> element to represent the virtual sentence formed by the aggregation of `s1` and `s2`:

```
<join result=s targets='qs3 qs4' >
```

As a further example, consider the following list of authors' names. The object of the <join> element here is to provide another list, composed of those authors from the larger list who happen to come from Heidelberg:

```
<list><head>Authors</head>
  <item id=uf>Figge, Udo
  <item id=ch>Heibach, Christiane
  <item id=gh>Heyer, Gerhard
  <item id=bp>Philipp, Bettina
  <item id=ms>Samiec, Monika
  <item id=ss>Schierholz, Stefan
</list>

<join result=list targets='ch bp ss' >
```

The following example shows how <join> can be used to reconstruct a text cited in fragments presented out of order. The poem being remembered (an unusual translation of a well known poem by Basho) runs "When the old pond / gets a new frog, / it's a new pond."

```
<sp who='Hughie'><p>How does it go?
  <q><l id=X1>da-da-da
     <l id=L2>gets a new frog
     <l>...
  </q></p>
<sp who='Louie'><p><q><l id=L1>When the old pond</l> ...</q></p>
<sp who='Dewey'><p><q>... <l id=L3>It's a new pond.</l></q></p>
<!-- ... -->
<join result=lg targets='L1 L2 L3' >
```

As with other forms of link, a grouping element <joinGrp> is available for use when a number of <join> elements of the same kind co-occur. This avoids the need to specify the result attribute for each <join> if they are all of the same type, and also allows us to restrict the domain within which their target elements are to be found, in the same way as for <linkGrp> elements (see 14.1.3 ('Groups of Links') on p. 337). Like a <join>, a <joinGrp> may appear only where the elements represented by its contents are legal. Thus if we had created many <join> tags of the sort just described, we could group them together, and require that their components are all contained by an element with the identifier MFKFhungry as follows:

```
<joinGrp result=s
         targType=s
         domains=MFKFhungry>
<!-- ... -->
  <join targets='qs3 qs4'>
  <join targets='qs5 qs6'>
  </joinGrp>
<!-- ... -->
```

The <join> element is useful as a means of representing non-hierarchic structures (as further discussed in chapter 31 ('Multiple Hierarchies') on p. 633). It may also be used as a convenient way of representing a variety of analytic units, like the and <interp> elements discussed in chapter 15 ('Simple Analytic Mechanisms') on p. 381. As an example, consider the following passage:

"

Zui-Gan called out to himself every day, "Master."

Then he answered himself, "Yes, sir."

And then he added, "Become sober."

Again he answered, "Yes, sir."

"And after that," he continued, "do not be deceived by others."

"Yes, sir; yes, sir," he replied."

Suppose now that we wish to represent an interpretation of the above passage in which we distinguish between the various "voices" adopted by the character Zui-Gan. In the following encoding, the who attribute has been used for this purpose; id attributes have also been added:

```
<text id=zuitxt>
<body>
<p>Zui-Gan called out to himself every day,
<q id=zuiq1 who=zuigan>Master.</q></p>
<p>Then he answered himself,
<q id=zuiq2 who=zuigan>Yes, sir.</q></p>
<p>And then he added,
<q id=zuiq3 who=master>Become sober.</q></p>
<p>Again he answered,
<q id=zuiq4 who=zuigan>Yes, sir.</q>
<p><q id=zuiq5 who=master>And after that,</q>
he continued,
<q id=zuiq6 who=master>do not be deceived by others.</q>
<p><q id=zuiq7 who=zuigan>Yes, sir; yes, sir,</q>
```

```
he replied.</p>
</body>
</text>
```

The id values specified now allow us to link the material spoken by each voice:

```
<text id=zuitxt>
<body>
<p>Zui-Gan called out to himself every day,
<q id=zuiq1 who=zuigan next=zuiq2>Master.</q></p>
<p>Then he answered himself,
<q id=zuiq2 who=zuigan next=zuiq4>Yes, sir.</q></p>
<p>And then he added,
<q id=zuiq3 who=master next=zuiq5>Become sober.</q></p>
<p>Again he answered,
<q id=zuiq4 who=zuigan next=zuiq7>Yes, sir.</q>
<p><q id=zuiq5 who=master next=zuiq6>And after that,</q>
he continued,
<q id=zuiq6 who=master>do not be deceived by others.</q>
<p><q id=zuiq7 who=zuigan>Yes, sir; yes, sir,</q>
he replied.</p>
</body>
</text>
```

However, by using the <join> element, we can directly represent the complete speech attributed to each voice:

```
<joinGrp targType='q' domains='zuitxt' result='q'>
  <join desc="what Zui-Gan said" targets='zuiq1 zuiq2 zuiq4 zuiq7'>
  <join desc="what Master said"  targets='zuiq3 zuiq5 zuiq6'>
  </joinGrp>
```

Note the use of the global n attribute to supply a descriptive name to distinguish the two virtual <q> elements represented by the <join> elements; this is necessary because the current proposals do not allow for any way of specifying the attributes to be associated with a virtual element, and hence we cannot specify a who value for them.

Suppose now that id attributes, for whatever reasons, are not available. Then <xptr> elements may be created using any of the methods described in section 14.2 ('Extended Pointers') on p. 340. The id attributes of *these* elements may now be specified by the targets attribute on the <join> elements.

```
<text><body>
<!-- ... -->
<!-- We are in the second DIV0, the sixth DIV1 ... -->
<p>Zui-Gan called out to himself every day,
   <q>Master.</q></p>
<p>Then he answered himself, <q>Yes, sir.</q></p>
<p>And then he added, <q>Become sober.</q></p>
<p>Again he answered, <q>Yes, sir.</q>
<p><q>And after that,</q> he continued,
<q>do not be deceived by others.</q></p>
<p><q>Yes, sir; yes, sir,</q> he replied.</p>
<!-- ... -->
</body></text>

<!-- ... -->

<xptr id=rzuiq1 from='DESCENDANT (2 div0) (6 div1) (1 p) (1 q)'>
<xptr id=rzuiq2 from='DESCENDANT (2 div0) (6 div1) (2 p) (1 q)'>
<xptr id=rzuiq3 from='DESCENDANT (2 div0) (6 div1) (3 p) (1 q)'>
<xptr id=rzuiq4 from='DESCENDANT (2 div0) (6 div1) (4 p) (1 q)'>
<xptr id=rzuiq5 from='DESCENDANT (2 div0) (6 div1) (5 p) (1 q)'>
<xptr id=rzuiq6 from='DESCENDANT (2 div0) (6 div1) (5 p) (2 q)'>
<xptr id=rzuiq7 from='DESCENDANT (2 div0) (6 div1) (6 p) (1 q)'>
```

```
<!-- ... -->

<joinGrp targType='xptr' result='q' evaluate=Y>
  <join desc="what Zui-Gan said" targets='rzuiq1 rzuiq2 rzuiq4 rzuiq7'>
  <join desc="what Master said"  targets='rzuiq3 rzuiq5 rzuiq6'>
</joinGrp>
```

For a definition of the syntax used by the <xptr> element, see section 14.2.2 ('Extended Pointer Syntax') on p. 341 above. The extended pointer with identifier rzuiq2 (for example) may be read as "the first <q> in the first <p>, inside the sixth <div1> within the second <div0> element of the current document."

As mentioned above, there is no need for the <join> and <xptr> elements to be held in the same SGML document as the text; indeed, if, for example, the text is held on a read-only medium, this may not be possible. The doc attribute of the <xptr> element may be used to specify the name of the SGML entity within which its target is to be found.

```
<!-- ... -->
<!ENTITY fazdoc SYSTEM 'C:\faz.doc'>
<!-- ... -->
<xptr id=rzuiq1
      doc=fazdoc
      from='DESCENDANT (2 div0) (6 div1) (1 p) (1 q)'>
<!-- ... -->
```

Here are the formal declarations of the <join> and <joinGrp> elements.

```
<!-- 14.7:  Aggregation                                        -->
<!ELEMENT join          - O  EMPTY                              >
<!ATTLIST join               %a.global;
          type               CDATA               #IMPLIED
          resp               CDATA               #IMPLIED
          crdate             CDATA               #IMPLIED
          targType           NAMES               #IMPLIED
          evaluate           (all | one | none)  #IMPLIED
          targets            IDREFS              #REQUIRED
          result             NAME                %INHERITED
          desc               CDATA               %INHERITED
          scope              (root | branches)   root
          targOrder          (Y | N | U)         Y             >
<!ELEMENT joinGrp       - -  ((join | ptr | xptr)*)             >
<!ATTLIST joinGrp            %a.global;
                             %a.pointerGroup;
          result             CDATA               #IMPLIED
          desc               CDATA               #IMPLIED      >
<!-- This fragment is used in sec. 14                          -->
```

14.8 Alternation

This section proposes elements for the representation of alternation. We say that two or more elements are in *exclusive alternation* if any of those elements could be present in a text, but one and only one of them is; in addition, we say that those elements are *mutually exclusive*. We say that the elements are in *inclusive alternation* if at least one (and possibly more) of them is present. The elements that are in alternation may also be called *alternants*.

The need to mark exclusive alternation arises frequently in text encoding. A common situation is one in which it can determined that exactly one of several different words appears in a given location, but it cannot be determined which one. One way to mark such an exclusive alternation is to use the linking attribute exclude. Having marked an exclusive alternation, it can sometimes later be determined which of the alternants actually appears in the given location. To

preserve the fact that an alternation was posited, one can add the linking attribute select to a tag which hierarchically encompasses the alternants, which points to the one which actually appears. To assign responsibility and degree of certainty to the choice, one can use the <certainty> tag described in chapter 17 ('Certainty and Responsibility') on p. 435. Also see that chapter for further discussion of certainty in general.

The exclude and select attributes may be used with any element assuming that they have been declared following the procedure discussed in the introduction to this chapter.

exclude points to elements that are in exclusive alternation with the current element.

select selects one or more alternants; if one alternant is selected, the ambiguity or uncertainty is marked as resolved. If more than one alternant is selected, the degree of ambiguity or uncertainty is marked as reduced by the number of alternants not selected.

A more general way to mark alternation, encompassing both exclusive and inclusive alternation, is to use the linking element <alt>. The description and attributes of this tag and of the associated grouping tag <altGrp> are as follows. These elements are also members of the *pointer* class and therefore have all the attributes associated with that class.

<alt> identifies an alternation or a set of choices among elements or passages. Attributes include:

- **targets** specifies the SGML identifiers of the alternative elements or passages.
- **weights** If mode=excl, each weight states the probability that the corresponding alternative occurs. If mode=incl each weight states the probability that the corresponding alternative occurs given that at least one of the other alternatives occurs.

<altGrp> groups a collection of <alt> elements and possibly pointers.

To take a simple hypothetical example, suppose in transcribing a spoken text, we encounter an utterance that we can understand either as 'We had fun at the beach today.' or as 'We had sun at the beach today.' We can represent the exclusive alternation of these two possibilities by means of the exclude attribute as follows.

```
<div>
<!-- ... -->
<u id=we.fun exclude=we.sun>We had fun at the beach today.</u>
<u id=we.sun exclude=we.fun>We had sun at the beach today.</u>
<!-- ... -->
</div>
```

If it is then determined that the speaker said 'fun', not 'sun', the encoder could amend the text by deleting the alternant containing 'sun' and the exclude attribute on the remaining alternant. Alternatively, the encoder could preserve the fact that there was uncertainty in the original transcription by retaining the alternants, and assigning the select=we.fun attribute value to the tag that encompasses the alternants, as in:

```
<div select=we.fun>
<!-- ... -->
<u id=we.fun exclude=we.sun>We had fun at the beach today.</u>
<u id=we.sun exclude=we.fun>We had sun at the beach today.</u>
<!-- ... -->
</div>
```

The above alternation (including the select attribute) could be recoded by assigning the exclude attributes to tags that enclose just the words or even the characters that are mutually exclusive, as in:[12]

```
<div>
<!-- ... -->
<u select=fun>We had
<seg type=word id=fun exclude=sun>fun</seg>
<seg type=word id=sun exclude=fun>sun</seg>
```

[12] See section 15.1 ('Linguistic Segment Categories') on p. 382 for discussion of the <w> and <c> tags that can be used in the following examples instead of the <seg type=word> and <seg type=character> tags.

```
at the beach today.</u>
<!-- ... -->
</div>

<div>
<!-- ... -->
<u>We had
<seg type=word select=f>
<seg type=character id=f exclude=s>f</seg>
<seg type=character id=s exclude=f>s</seg>
un</seg>
at the beach today.
</u>
<!-- ... -->
</div>
```

Now suppose that the transcriber is uncertain whether the first word in the utterance is 'We' or 'Lee', but is certain that if it is 'Lee', then the other uncertain word is definitely 'fun' and not 'sun'. The three utterances that are in mutual exclusion can be encoded as follows.

```
<div>
<!-- ... -->
<u id=we.fun exclude='we.sun lee.fun'>
We had fun at the beach today.</u>
<u id=we.sun exclude='we.fun lee.fun'>
We had sun at the beach today.</u>
<u id=lee.fun exclude='we.fun we.sun'>
Lee had fun at the beach today.</u>
<!-- ... -->
</div>
```

The preceding example can also be encoded with `exclude` attributes on the word segments 'We', 'Lee', 'fun' and 'sun':

```
<u>
<seg type=word id=we exclude=lee>We</seg>
<seg type=word id=lee exclude='we sun'>Lee/seg>
had
<seg type=word id=fun exclude=sun>fun</seg>
<seg type=word id=sun exclude='fun lee'>sun</seg>
at the beach today.</u>
```

The value of the `select` attribute is defined as a list of identifiers (`IDREFS`); hence it can also be used to narrow down the range of alternants, as in:

```
<div select='we.fun lee.fun'>
<!-- ... -->
<u id=we.fun exclude='we.sun lee.fun'>
We had fun at the beach today.</u>
<u id=we.sun exclude='we.fun lee.fun'>
We had sun at the beach today.</u>
<u id=lee.fun exclude='we.fun we.sun'>
Lee had fun at the beach today.</u>
<!-- ... -->
</div>
```

This is interpreted to mean that either the first or the third `<u>` tag appears, and is thus equivalent to just the alternation of those two tags:

```
<div>
<!-- ... -->
<u id=we.fun exclude=lee.fun>We had fun at the beach today.</u>
<u id=lee.fun exclude=we.fun>Lee had fun at the beach today.</u>
<!-- ... -->
</div>
```

The exclude attribute can also be used in case there is uncertainty about the tag that appears in a certain position. For example, the occurrence of the word 'May' in the S-unit 'Let's go to May' can be interpreted, in the absence of other information, either as a person's name or as a date. The uncertainty can be rendered as follows, using the exclude attribute.

```
<s>Let's go to
<name id=mayd exclude=mayn>May</name>
<date id=mayn exclude=mayd copyOf=mayd></date>.</s>
```

Note the use of the copyOf attribute discussed in section 14.6 ('Identical Elements and Virtual Copies') on p. 367; this avoids having to repeat the content of the element whose correct tagging is in doubt.

The copyOf and the exclude attributes also provide for a simple way of indicating uncertainty about exactly where a particular element occurs in a document.[13] For example suppose that a particular <div2> element appears either as the third and last of the <div2> elements within the first <div1> element in the body of a document, or as the first <div2> of the second <div1>. One solution would be to record the <div2> in its entirety in the first of these positions, and a virtual copy of it in the second, and mark them as excluding each other as follows:

```
<body>
<div1 id=c1>

 <div2 id=c1s3 exclude=c2s1>
  <!-- Text of the "movable" div2 appears here.  -->
 </div2>

</div1>

<div1 id=c2>

 <div2 id=c2s1 copyOf=c1s3 exclude=c1s3></div2>

</div1>

</body>
```

In this case, the select attribute, if used, would appear on the <body> tag.

Mutual exclusion can also be expressed using a <link>; the first example in this section can be recoded by removing the exclude attributes from the <u> tags, and adding a <link> as follows:[14]

```
<div>
<!-- ... -->
<u id=we.fun>We had fun at the beach today.</u>
<u id=we.sun>We had sun at the beach today.</u>
<link type='exclusive alternation'
      targType='u u'
      targOrder=N
      targets='we.fun we.sun'>
<!-- ... -->
</div>
```

Now we define the specialized linking element <alt>,making it a member of the pointer class of elements, and assigning it a excl (for 'mutually exclusive') attribute, which can have either of the values Y or N. Then the following equivalence holds:

```
<alt mode=excl> = <link type='exclusive alternation'>
```

It is in the nature of alternation that the order of the targets is irrelevant; hence the targOrder attribute of the <alt> defaults to the value N. The preceding <link> may therefore be recoded as the following <alt> tag.

[13] An alternative way of representing this problem is discussed in chapter 17 ('Certainty and Responsibility') on p. 435.

[14] In this example, we have placed the <link> next to the tags that represent the alternants. It could also have been placed elsewhere in the document, perhaps within a <linkGrp>.

```
<alt mode=excl targType='u u' targets='we.fun we.sun'>
```

Other attributes that are defined specifically for the <alt> element are weights and percent. The weights attribute is to be used if one wishes to assign *probabilistic weights* to the targets (alternants). Its value is a list of numbers, corresponding to the targets, expressing the probability that each target appears. The percent attribute is used to indicate whether the weights are stated as percentages (percent=Y, the default) or as the actual probabilities (percent=N). If the alternants are mutually exclusive, then the weights must sum to 100% (or 1, if percent=N is specified).

Suppose in the preceding example that it is equiprobable whether 'fun' or 'sun' appears. Then the <alt> that represents the alternation may be stated as follows:

```
<alt mode=excl
     targType='u u'
     targets='we.fun we.sun'
     weights='50 50'>
```

The assignment of a weight of 100% to one target (and weights of 0% to all the others) is equivalent to selecting that target. Thus the following encoding is equivalent to the second example at the beginning of this section.

```
<div>
<!-- ... -->
<u id=we.fun>We had fun at the beach today.</u>
<u id=we.sun>We had sun at the beach today.</u>
<alt mode=excl
     targType='u u'
     targets='we.fun we.sun'
     weights='100 0'>
<!-- ... -->
</div>
```

The sum of the weights for <alt excl=N> tags ranges from 0% to (100 x k)%, where k is the number of targets. If the sum is 0%, then the alternation is equivalent to exclusive alternation; if the sum is (100 x k)%, then all of the alternants must appear, and the situation is better encoded without an <alt> tag.

If it is desired, <alt> tags may be grouped together in an <altGrp> tag, and attribute values shared by the individual <alt> tags may be identified on the <altGrp> tag. The targFunc attribute defaults to the value `'first.alternant next.alternant'`. Thus, specifying the extendTarg=2 attribute value permits the alternants to be extended indefinitely.

To illustrate, consider again the example of a transcribed utterance, in which it is uncertain whether the first word is 'We' or 'Lee', whether the third word is 'fun' or 'sun', but that if the first word is 'Lee', then the third word is 'fun'. Now suppose we have the following additional information: if 'we' occurs, then the probability that 'fun' occurs is 50% and that 'sun' occurs is 50%; if 'fun' occurs, then the probability that 'we' occurs is 40% and that 'Lee' occurs is 60%. This situation can be encoded as follows.

```
<u>
<seg type=word id=we exclude=lee>We</seg>
<seg type=word id=lee exclude=we>Lee/seg>
had
<seg type=word id=fun exclude=sun>fun</seg>
<seg type=word id=sun exclude=fun>sun</seg>
at the beach today.</u>
<!-- ... -->
<altGrp targType='seg seg'>
  <alt targets='we lee'>
  <alt targets='fun sun'>
  <alt mode=incl targets='we fun'  weights='50  50'>
  <alt mode=incl targets='Lee fun' weights='100 60'>
  </altGrp>
```

From the information in this encoding, we can determine that the probability is about 28.5%

that the utterance is "We had fun at the beach today", 28.5% that it is 'We had sun at the beach today', and 43% that it is 'Lee had fun at the beach today'.

Another very similar example is the following regarding the text of a Broadway song. In three different versions of the song, the same line reads "Her skin is tender as a leather glove," "Her skin is tender as a baseball glove," and "Her skin is tender as Dimaggio's glove."[15]

If we wish to express this textual variation using the <alt> element, we can record our relative confidence in the readings 'Dimaggio's' (with probability 50%), 'a leather' (25%), and 'a baseball' (25%).

Let us extend the example with a further (imaginary) variation, supposing for the sake of the argument that the next line is variously given as 'and she bats from right to left' (with probability 50%) or 'now ain't that too damn bad' (with probability 50%). Using the <alt> element, we can express the conviction that if the first choice for the second line is correct, then the probability that the first line contains 'Dimaggio's' is 90%, and each of the others 5%; whereas if the second choice for the second line is correct, then the probability that the first line contains 'Dimaggio's' is 10%, and each of the others is 45%. This can be encoded, with an <altGrp> tag containing a combination of exclusive and inclusive <alt> tags, as follows.

```
<div id=bm>
<!-- ... -->
<l>Her skin is tender as
<seg id=dm>Dimaggio's</seg>
<seg id=lt>a leather</seg>
<seg id=bb>a baseball</seg>
glove,</l>
<l id=rl>and she bats from right to left.</l>
<l id=db>now ain't that too damn bad.</l>
<!-- ... -->
</div>
<!-- ... -->
<altGrp>
  <alt mode=excl targType='seg seg'
       targets='dm lt bb' weights='50 25 25'>
  <alt mode=Excl targType='l l'
       targets='rl db' weights='50 50'>
  </altGrp>
<altGrp mode=incl targType='seg l'>
  <alt targets='dm rl' weights='90 90'>
  <alt targets='lt rl' weights='5 5'>
  <alt targets='bb rl' weights='5 5'>
  <alt targets='dm db' weights='10 10'>
  <alt targets='lt db' weights='45 90'>
  <alt targets='bb db' weights='45 90'>
  </altGrp>
```

Here are the formal declarations of the <alt> and <altGrp> elements.

```
<!-- 14.8:  Alternation                                          -->
<!ELEMENT alt           - O  EMPTY                                 >
<!ATTLIST alt                %a.global;
          type               CDATA               #IMPLIED
          resp               CDATA               #IMPLIED
          crdate             CDATA               #IMPLIED
          targType           NAMES               #IMPLIED
          evaluate           (all | one | none)  #IMPLIED
          targets            IDREFS              #REQUIRED
          mode               (excl | incl)       %INHERITED
          weights            NUMBERS             #IMPLIED
          wScale             (perc | real)       %INHERITED
          targOrder          (Y | N)             N              >
```

[15]The variant readings are found in the commercial sheet music, the performance score, and the Broadway cast recording.

```
<!ELEMENT altGrp        - -  ((alt | ptr | xptr)*)              >
<!ATTLIST altGrp             %a.global;
                             %a.pointerGroup;
          mode               (excl | incl)       excl
          wScale             (perc | real)       perc          >
<!-- This fragment is used in sec. 14                          -->
```

14.9 Connecting Analytic and Textual Markup

In chapters 15 (‘Simple Analytic Mechanisms’) on p. 381 and 16 (‘Feature Structures’) on p. 397 and elsewhere, provision is made for analytic and interpretive markup to be represented outside of textual markup, either in the same document or in a different document. The elements in these separate domains can be connected, either with the pointing attributes ana (for ‘analysis’) and inst (for ‘instance’), or by means of <link> and <linkGrp> elements. Numerous examples are given in these chapters, particularly in sections 15.4 (‘Linguistic Annotation’) on p. 392, 16.3 (‘Feature, Feature-Structure and Feature-Value Libraries’) on p. 400 and 16.10 (‘Two Illustrations’) on p. 427.

Chapter 15

Simple Analytic Mechanisms

This chapter describes a tag set for associating simple analyses and interpretations with text elements. We use the term *analysis* here to refer to any kind of semantic or syntactic interpretation which an encoder wishes to attach to all or part of a text. Examples discussed in this chapter include familiar linguistic categorizations (such as "clause", "morpheme", "part-of-speech" etc.) and characterizations of narrative structure (such as "theme", "reconciliation" etc.). The mechanisms presented in this chapter offer simpler but less powerful than those described in chapter 16 ('Feature Structures') on p. 397.

Section 15.1 ('Linguistic Segment Categories') on p. 382 introduces a tag set for characterizing text segments according to the familiar linguistic categories of *sentence* or *s-unit*, *clause*, *phrase*, *word*, *morpheme*, and *character*. These elements represent special cases of the generic <seg> element described in section 14.3 ('Segments and Anchors') on p. 355.

Section 15.2 ('Global Attributes for Simple Analyses') on p. 387 introduces an additional global attribute which allows passages of text to be associated with specialised SGML elements representing their interpretation. These "interpretative" elements (and <interp>) are described in detail in section 15.3 ('Spans and Interpretations') on p. 387. They allow the encoder to specify an analysis as a series of names and associated values,[1] each such pair being linked to one or more stretches of text, either directly, in the case of spans, or indirectly, in the case of interpretations.

Finally section 15.4 ('Linguistic Annotation') on p. 392 revisits the topic of linguistic analysis, and illustrates how these interpretative mechanisms may be used to associate simple linguistic analysis with text segments.

The following DTD fragments show the overall organization of the class of analytic elements discussed in the remainder of this chapter. File *teiana2.ent* defines the additional global attribute made available by this tag set.

```
<!-- 15:  Modifications to TEI class system for analysis      -->
<!-- ... declarations from section 15.2                       -->
<!--     (Global attribute for analysis)                      -->
<!--     go here ...                                          -->
```

File *teiana2.dtd* contains declarations for elements used to represent simple analyses or interpretations of portions of a text.

```
<!-- 15:  Simple analytic mechanisms                          -->
 <!-- Text Encoding Initiative: Guidelines for Electronic      -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.    -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy    -->
<!-- in any form is granted, provided this notice is          -->
<!-- included in all copies.                                  -->

<!-- These materials may not be altered; modifications to     -->
```

[1] Or, as they are widely known, *attribute-value pairs*; this term should not be confused, however, with SGML attributes and their values, which are similar in concept but distinct in their formal definitions.

```
<!-- these DTDs should be performed as specified in the       -->
<!-- Guidelines in chapter "Modifying the TEI DTD."           -->

<!-- These materials subject to revision. Current versions    -->
<!-- are available from the Text Encoding Initiative.         -->

<!-- We declare the various elements, group by group.         -->

<!-- ... declarations from section 15.3                       -->
<!--     (Spans)                                              -->
<!--     go here ...                                          -->
<!-- ... declarations from section 15.1                       -->
<!--     (Linguistic Segment Categories)                      -->
<!--     go here ...                                          -->
```

This tag set is selected as described in 3.3 ('Invocation of the TEI DTD') on p. 39; in a document which uses the markup described in this chapter, the document type declaration should contain the following declaration of the entity *TEI.analysis*, or an equivalent one:

```
<!ENTITY % TEI.analysis 'INCLUDE'>
```

The entire document type declaration for a document using this additional tag set together with that for linking and alignment and the base tag set for prose might look like this:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
   <!ENTITY TEI.prose    'INCLUDE' >
   <!ENTITY TEI.linking  'INCLUDE' >
   <!ENTITY TEI.analysis 'INCLUDE' >
]>
```

15.1 Linguistic Segment Categories

In this section we introduce specialized *linguistic segment category* elements which may be used to represent the segmentation of a text into the traditional linguistic categories of *sentence*, *clause*, *phrase*, *word*, *morpheme*, and *characters*.

<s> contains a sentence-like division of a text.
<cl> represents a grammatical clause.
<phr> represents a grammatical phrase.
<w> represents a grammatical (not necessarily orthographic) word. Attributes include:

lemma identifies the word's lemma (dictionary entry form).

<m> represents a grammatical morpheme. Attributes include:

baseform identifies the morpheme's base form.

<c> represents a character.

As members of the *seg* class, these elements share the following attributes:

type characterizes the type of segment.
function characterizes the function of the segment.

The **<s>** element may be used simply to segment a text end-to-end into a series of non-overlapping segments, referred to here and elsewhere as *s-units*, or *sentences*.

```
<p>
<s>Nineteen fifty-four, when I was eighteen years old,
is held to be a crucial turning point in the history of
the Afro-American  --- for the U.S.A. as a whole --- the
year segregation was outlawed by the U.S. Supreme Court.</s>
<s>It was also a crucial year for me because on June 18,
1954, I began serving  a sentence in state prison for
possession of marijuana.</s>
```

The <s>may be thought of as providing an abbreviated version of the tag <seg type='s-unit'>, with the important additional proviso that (unlike <seg> elements) <s> elements may not be nested within each other. The type attribute of the <s> element corresponds to the subtype attribute on the <seg> element, that is, a tag <s type='xxx'> should be thought of as synonymous with a tag <seg type='s-unit' subtype='xxx'>. Similar considerations apply to the <cl> and <phr> elements , which can be thought of as short for <seg type=clause> and <seg type=phrase>, respectively.

The <s> element may be further subdivided into *clauses*, marked with the <cl> element, as in the following example:

```
<p>
<s><cl>It was about the beginning of September, 1664,
    <cl>that I, among the rest of my neighbours,
      heard in ordinary discourse
      <cl>that the plague was returned again
        in Holland;
      </cl>
    </cl>
  </cl>
  <cl>for it had been very violent there, and
    particularly at Amsterdam and Rotterdam,
    in the year 1663,</cl>
  <cl>whither,
    <cl>they say,</cl>
    it was brought,
    <cl>some said</cl>
    from Italy, others from the Levant,
    among some goods
    <cl>which were brought home
    by their Turkey fleet;</cl>
  </cl>
  <cl>others said it was brought from Candia;
    others from Cyprus.
  </cl>
</s>
<s><cl>It mattered not
    <cl>from whence it came;</cl>
  </cl>
  <cl>but all agreed
    <cl>it was come into Holland again.</cl>
  </cl>
</s>
</p>
```

Clauses may be further divided into <phr> elements in the same way. A text may be segmented directly into clauses, or into phrases, with no need to include segmentation at a higher level as well.

For verse texts, the overlapping of metrical and syntactic structure requires that special care be given to representing both using the SGML element hierarchy. One simple approach is to split the syntactic phrases into fragments when they cross verse boundaries, reuniting them either with the part attribute:

```
<div type=stanza>
<l><cl part=i>Tweedledum and Tweedledee</cl></l>
<l><cl part=f>Agreed to have a battle;</cl></l>
<l><cl part=i>For Tweedledum said
  <cl part=i>Tweedledee</cl></cl></l>
<l><cl part=f><cl part=f>Had spoiled his
  nice new rattle.</cl></cl></l>
<div type=stanza>
<l><cl part=i>Just then flew down a monstrous crow,</cl></l>
<l><cl part=f>As black as a tar barrel;</cl></l>
```

```
<l><cl part=i>Which frightened both the heroes so,</cl></l>
<l><cl part=f><cl>They quite forgot their quarrel.</cl></cl></l>
</div>
```

Another approach is to use the next and prev attributes defined in the additional tag set for linking (chapter 14 ('Linking, Segmentation, and Alignment') on p. 331):

```
<div type=stanza>
<l>...
<l><cl id=c3 next=c5 part=i>For Tweedledum said
  <cl id=c4 next=c6 part=i>Tweedledee</cl></cl></l>
<l><cl id=c5 prev=c3 part=f>
  <cl id=c6 prev=c4 part=f>
   Had spoiled his nice new rattle.</cl></cl></l>
</div>
```

Other methods are also possible; for discussion, see chapter 31 ('Multiple Hierarchies') on p. 633.

The type attribute on linguistic segment categories can be used to provide additional interpretative information about the category. The function attribute on the <cl> and <phr> elements can be used to provide additional information about the function of the category. Legal values for these two attributes are not defined by these Guidelines, but should be documented in the <segmentation> element of the <encodingDesc> element within the document's header. A general approach to the encoding of linguistic categories assigned to parts of a text is discussed in section 15.4 ('Linguistic Annotation') on p. 392 below.

Using traditional terminology, these attributes provide a convenient way of specifying, for example, that the clause 'from whence it came' is a relative clause modifying another, or that the phrase 'by the U.S. Supreme Court' is a prepositional post-modifier:

```
<cl>It mattered not
  <cl type='relative' function='clause modifier'>
   from whence it came;</cl></cl>

<phr type=NP>the year
  segregation</phr>
 <phr>was outlawed</phr>
 <phr type=PP function='postmodifier (agent)'>
  by the U.S. Supreme Court</phr>
</phr>.
```

Segmentation into clauses and phrases can, of course, be combined. To make such encodings easier to read, the segmentation tags can begin new lines, and be indented according to their degree of nesting, thus:

```
<p>
<cl type='finite declarative' function='independent'>
 <phr type=NP function='subject'>Nineteen fifty-four,
  <cl type='finite relative declarative'
    function='appositive'>when
   <phr type=NP function='subject'>I</phr>
   <phr type=VP function='predicate'>was eighteen
                    years old</phr>
  </cl>,
 </phr>
 <phr type=VP function='predicate'>
  <phr type=V function='main verb'>is held</phr>
  <phr type=NP function='complement'> <!-- ? -->
   <cl type='nonfinite' function='predicate nom.'>
    <phr type=V function='copula'>to be</phr>
    <phr type=NP function='predicate nom.'>
     a crucial turning point
     <phr type=PP function='postmodifier'>in
      <phr type=NP function='prep.obj.'>
```

```
       the history
       <phr type=PP function='postmodifier'>
        of the Afro-American</phr>
      </phr>
     </phr>
     ---
     <phr type=PP
        function='appositive postmodifier'>for
      <phr type=NP function='prep.obj.'>the U.S.A.
       <phr type=PP function='postmodifier'>
        as a whole</phr>
      </phr>
     </phr>
    </phr>
    ---
    <phr type=NP function='appositive pred.nom.'>
     the year
     <cl type='finite relative'
       function='adjectival'>
      <phr type=NP function='subject'>
       segregation</phr>
      <phr type=VP function='predicate'>
       <phr type=V function='main verb'>
        was outlawed</phr>
       <phr type=PP function='postmodifier'>
        by the U.S. Supreme Court</phr>
      </phr>
     </cl>
    </phr>
   </cl>
  </phr>
 </phr>
  .
</cl>
<cl type='finite declarative' function='independent'>
 <phr type=NP function='subject'>It</phr>
 <phr type=VP function='predicate'>
  <phr type=V function='main verb'>was</phr>
  also
  <phr type=NP function='predicate nom.'>
   a crucial year for me</phr>
 </phr>
 <cl type='finite declarative'
   function='dependent causative'>because
  <phr type=PP function='sentence adverb'>
   on June 18, 1954</phr>,
  <phr type=NP function='subject'>I</phr>
  <phr type=VP function='predicate'>
   <phr type=V function='main verb'>began serving</phr>
   <phr type=NP function='complement'>
    a sentence in state prison
    <phr type=PP function='complement'>
     for possession of marijuana
    </phr>
   </phr>
  </phr>
 </cl>
</cl>
 .
```

This style of markup, however, introduces spurious new lines and blanks into the text, which could make restoring the text to its original layout problematic. If the original layout is important, the original line breaks and font shifts should be recorded using <lb> elements, the global rend attribute, etc.

The <w>, <m> and <c> elements are also identical in meaning to the <seg> element with a type attribute of "w", "m", or "c", and may occur wherever <seg> is permitted to occur. However, they have more restricted content models than does <seg>: for example, the <w> element can only contain <w>, <m> and <c> elements, and parsed character data; the <m> element can only contain <c> elements and parsed character data; the <c> element can only contain parsed character data, and should in fact only contain a single character. Consequently, while <m> et al. can be translated directly into typed <seg> elements, the reverse is not necessarily the case.

The restriction on the content of the <w> attribute in particular requires that a certain care must be exercised when using it, especially in relation to the use of other tags that one may think of as *word level*, but which are in fact defined as *phrase level*. Consider the problem of segmenting an occurrence of the <mentioned> element as a word.

```
<mentioned>grandiloquent</mentioned>
```

The first of the following two encodings is legitimate; the second is not, since the <mentioned> element is not part of the content model of the <w> element:

```
<!-- This is all right. -->
<mentioned><w>grandiloquent</w></mentioned>

<!-- This is NOT all right! -->
<w><mentioned><grandiloquent</mentioned></w>
```

On the other hand, both of the following encodings *are* legitimate:

```
<mentioned><phr>grandiloquent speech</phr></mentioned>

<phr><mentioned>grandiloquent speech</mentioned></phr>
```

The first encoding describes the citing of a phrase. The second describes a phrase which consists of something mentioned.

The <w> and <m> attributes carry additional attributes which may be of use in many indexing or analytic applications. The lemma attribute may be used to specify the *lemma*, that is the head- or base- form of an inflected verb or noun, for example:

```
<s lang=LA>
  <w lemma=timeo>timeo
  <w lemma=danaii>Danaos
  <w lemma=et>et
  <w lemma=donum>dona</w>
  <w lemma=fero>ferentes</w>
```

Similarly, the baseform attribute may be specified for the <m> element, to indicate the "base form" of a transformed morpheme:

```
<w type=adjective>
  <m type=prefix baseform=con>com</m>
  <m type=root>fort</m>
  <m type=suffix>able</m>
</w>
```

The <w>, <m> and <c> elements can be used together to give a fairly detailed low-level grammatical analysis of text. For example, consider the following segmentation of the English S-unit 'I didn't do it'.

```
<w>I</w>
<w>
 <w>did</w>
 <m>n't</m>
 </w>
```

```
<w>do</w>
<w>it</w>
<c>.</c>
```

This segmentation, crude as it is, succeeds in representing the idea that 'did' occurs as a word inside the word 'didn't'. A further advantage of segmenting the text down to this level is that it becomes relatively simple to associate each such segment with a more detailed formal analysis. This matter is taken up in detail in section 15.4 ('Linguistic Annotation') on p. 392.

The <s>, <cl>, <phr>, <w>, <m>, and <c> elements are formally declared as follows:

```
<!-- 15.1:  Linguistic Segment Categories                    -->
<!ELEMENT s             - -  (%phrase.seq)        -(s)        >
<!ATTLIST s                  %a.global;
                             %a.seg;                          >
<!ELEMENT cl            - -  (%phrase.seq)                    >
<!ATTLIST cl                 %a.global;
                             %a.seg;                          >
<!ELEMENT phr           - -  (%phrase.seq)                    >
<!ATTLIST phr                %a.global;
                             %a.seg;                          >
<!ELEMENT w             - -  ((#PCDATA | seg | w | m | c)*)   >
<!ATTLIST w                  %a.global;
                             %a.seg;
          lemma              CDATA               #IMPLIED     >
<!ELEMENT m             - -  ((#PCDATA | seg | c)*)           >
<!ATTLIST m                  %a.global;
                             %a.seg;
          baseform           CDATA               #IMPLIED     >
<!ELEMENT c             - -  (#PCDATA)                        >
<!ATTLIST c                  %a.global;
                             %a.seg;                          >
<!-- This fragment is used in sec. 15                        -->
```

15.2 Global Attributes for Simple Analyses

When the tag set described by this chapter is selected, an additional attribute is defined for all elements:

ana indicates one or more elements containing interpretations of the element on which the ana attribute appears.

The ana attribute may be specified for any SGML element. Its effect is to associate the element with one or more others representing an analysis or interpretation of it. Its target should be one of the elements described in the section 15.3 ('Spans and Interpretations') on p. 387 below, or some other interpretative element such as <note>, on which see section 6.8 ('Notes, Annotation, and Indexing') on p. 152 or <fs>, on which see chapter 16 ('Feature Structures') on p. 397.

The ana attribute is formally declared as follows:

```
<!-- 15.2:  Global attribute for analysis                    -->
<!ENTITY % a.analysis '
          ana                IDREFS              #IMPLIED'    >
<!-- This fragment is used in sec. 15                        -->
```

15.3 Spans and Interpretations

The simplest mechanisms for attaching analytic notes in some structured vocabulary to particular passages of text are provided by the empty and <interp> elements, and their associated grouping elements <spanGrp> and <interpGrp>.

**** associates an interpretative annotation directly with a span of text. Attributes include:

value identifies the specific phenomenon being annotated.

from specifies the beginning of the passage being annotated; if not accompanied by a **to** attribute, then specifies the entire passage.

to specifies the end of the passage being annotated.

<spanGrp> collects together tags.

<interp> provides for an interpretative annotation which can be linked to a span of text. Attributes include:

value identifies the specific phenomenon being annotated.

<interpGrp> collects together <interp> tags.

These elements are all members of the class *interpret*, and thus share the following attributes:

resp indicates who is responsible for the interpretation.

type indicates what kind of phenomenon is being noted in the passage. Sample values include:

image identifies an image in the passage.

character identifies a character associated with the passage.

theme identifies a theme in the passage.

allusion identifies an allusion to another text.

(discourse type) specifies that the passage is of a particular discourse type.

inst points to instances of the analysis or interpretation represented by the current element.

The **type** and **value** attributes of the and <interp> elements may be used to associate an interpretive name, type, and value with a specific stretch (or *span*) of text. In the case of the element, the span of text being annotated is indicated by values of the **from** and **to** attributes, the value of each being a pointer. If the optional **to** attribute is omitted, the span consists just of the element pointed at by the obligatory **from** attribute. In the case of <interp> (see below), the span is indicated by a pointer from a <link> element or some similar mechanism. Here is an example of the element.

```
<p id=MQp1s2p114>
<s id=MQp1s2p114s1>There was certainly a definite point
at which the thing began.

<s id=MQp1s2p114s2>It was not; then it was suddenly
inescapable, and nothing could have frightened it
away.</s>

<s id=MQp1s2p114s3>There was a slow integration,
during which she, and the little animals, and the moving
grasses, and the sun-warmed trees, and the slopes of
shivering silvery mealies, and the great dome of blue
light overhead, and the stones of earth under her feet,
became one, shuddering together in a dissolution
of dancing atoms.</s>

<s id=MQp1s2p114s4>She felt the rivers under the ground
forcing themselves pain&shy;fully along her veins,
swelling them out in an unbearable pressure; her flesh
was the earth, and suffered growth like a ferment; and
her eyes stared, fixed like the eye of the sun.</s>

<s id=MQp1s2p114s5>Not for one second longer (if the
terms for time apply) could she have borne it; but then,
with a sudden movement forwards and out, the whole
process stopped; and <emph rend=italic>that</emph> was
<soCalled rend=dquo>the moment</soCalled> which it was
impossible to remember afterwards.</s>

<span resp=DTL
```

```
        value='the moment'
        from=MQp1s2p114s3
        to=MQp1s2p114s5>

<s id=MQp1s2p114s6>For during that space of time (which
was timeless) she understood quite finally her
smallness, the unimportance of humanity.</s>

<!-- ... -->
</p>
```

The element may, as in this example, be placed in the text near the textual span it is associated with, or it may be placed outside the text enclosed within a <spanGrp> element as follows.

```
<spanGrp resp=DTL>
  <span value='the moment' from=MQp1s2p114s3 to=MQp1s2p114s5>
```

As may be seen, the type attribute may be omitted in order to associate a span of text simply with a descriptive name.

Spans may also be used to represent the structural divisions assigned to the narrative by an interpreter. Consider the following narrative:

"

Sigmund, the son of Volsung, was a king in Frankish country. Sinfiotli was the eldest of his sons, the second was Helgi, the third Hamund. Borghild, Sigmund's wife, had a brother named —- But Sinfiotli, her stepson, and —- both wooed the same woman and Sinfiotli killed him over it.[2] And when he came home, Borghild asked him to go away, but Sigmund offered her weregild, and she was obliged to accept it. At the funeral feast Borghild was serving beer. She took poison, a big drinking horn full, and brought it to Sinfiotli. When Sinfiotli looked into the horn, he saw that poison was in it, and said to Sigmund "This drink is cloudy, old man." Sigmund took the horn and drank it off. It is said that Sigmund was hardy and that poison did him no harm, inside or out. And all his sons could tolerate poison on their skin. Borghild brought another horn to Sinfiotli, and asked him to drink, and everything happened as before. And a third time she brought him a horn, and reproachful words as well, if he didn't drink from it. He spoke again to Sigmund as before. He said "Filter it through your mustache, son!" Sinfiotli drank it off and at once fell dead.

Sigmund carried him a long way in his arms and came to a long, narrow fjord, and there was a small boat there and a man in it. He offered to ferry Sigmund over the fjord. But when Sigmund carried the body out to the boat, it was fully laden. The man said Sigmund should go around the fjord inland. The man pushed the boat out and then suddenly vanished.

King Sigmund lived a long time in Denmark in the kingdom of Borghild, after he married her. Then he went south to Frankish lands, to the kingdom he had there. Then he married Hiordis, the daughter of King Eylimi. Their son was Sigurd. King Sigmund fell in a battle with the sons of Hunding. And then Hiordis married Alf, the son of King Hialprec. Sigurd grew up there as a boy.

Sigmund and all his sons were tall and outstanding in their strength, their growth, their intelligence, and their accomplishments. But Sigurd was the most outstanding of all, and everyone who knows about the old days says he was the most outstanding of men and the noblest of all the warrior kings."

A structural analysis of this text, dividing it into narrative units in a pattern shared with other texts from the same literature, might look like this:

```
<p id=P1>
<s id=S1>Sigmund ... was a king in Frankish country.</s>
<s id=S2>Sinfiotli was the eldest of his sons.</s>
<s id=S3>Borghild, Sigmund's wife, had a brother ... </s>
<s id=S4a>But Sinfiotli ... wooed the same woman</s>
<s id=S4b>and Sinfiotli killed him over it.</s>
<s id=S5>And when he came home, ... she was obliged to accept it.</s>
```

[2]The rule marks spaces left for the missing name in the manuscript.

```
<s id=S6>At the funeral feast Borghild was serving beer.</s>
<s id=S7>She took poison ... and brought it to Sinfiotli.</s>
<!-- ... -->
<s id=S17>Sinfiotli drank it off and at once fell dead.</s>
<anchor id=nil1>
<p id=P2>Sigmund carried him a long way in his arms ... </p>
<p id=P3>King Sigmund lived a long time in Denmark ... </p>
<p id=P4>Sigmund and all his sons were tall ... </p>

<!-- ... -->

<span resp=TMA type='structural unit' value='introduction'
      from=S1 to=S3                                          >
<span resp=TMA type='structural unit' value='conflict'
      from=S4a                                               >
<span resp=TMA type='structural unit' value='climax'
      from=S4b                                               >
<span resp=TMA type='structural unit' value='revenge'
      from=S5 to=S17                                         >
<span resp=TMA type='structural unit' value='reconciliation'
      from=nil1                                              >
<span resp=TMA type='structural unit' value='aftermath'
      from=P2 to=P4                                          >
```

Note the use of an empty <anchor> element to provide a target for the "reconciliation" unit which is normally part of the narrative pattern but which is not realized in the text shown.

If groups of elements with the same resp or type are used, as in this example, they may be grouped together inside a <spanGrp> element, with the values of the common attribute(s) inherited from the higher element, as follows.

```
<spanGrp resp=TMA type='structural unit'>
 <span value='introduction'   from=S1   to=S3 >
 <span value='conflict'       from=S4a        >
 <span value='climax'         from=S4b        >
 <span value='revenge'        from=S5   to=S17>
 <span value='reconciliation' from=nil1       >
 <span value='aftermath'      from=P2   to=P4 >
 </spanGrp>
```

The same analysis may be expressed with the <interp> element instead of the element; this element provide attributes for recording an interpretive category and its value, as well as the identity of the interpreter, but does not itself indicate which passage of text is being interpreted; the same interpretive structures can thus be associated with many passages of the text. The association between text passages and <interp> elements must be made either by pointing from the text to the <interp> element with the ana attribute defined in section 15.2 ('Global Attributes for Simple Analyses') on p. 387, or by pointing at both text and interpretation from a <link> element, as described in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331.

To encode the first example above using <interp>, it is necessary to create a text element which contains — or corresponds to — the the third, fourth, and fifth orthographic sentences (S-units) in the paragraph. This can be done either with the seg element, described in 14.3 ('Segments and Anchors') on p. 355, or the join element, described in 14.7 ('Aggregation') on p. 369. The resulting element can then be associated with the <interp> element using the ana attribute described in section 15.2 ('Global Attributes for Simple Analyses') on p. 387. We illustrate using the <seg> element.

```
<p id=MQp1s2p114>
<s id=MQp1s2p114s1>There was certainly a definite point ... </s>
<s id=MQp1s2p114s2>It was not; then it was suddenly inescapable ... </s>
<seg id=MQp1s2p114s3-5 ana=moment>
<s id=MQp1s2p114s3>There was a slow integration ... </s>
<s id=MQp1s2p114s4>She felt the rivers under the ground ... </s>
```

```
<s id=MQp1s2p114s5>Not for one second longer ... </s>
</seg>
<s id=MQp1s2p114s6>For during that space of time ... </s>
<!-- ... -->
</p>
<!-- ... -->
<interp id=moment resp=DTL value='the moment'>
```

The second example above can be recoded using <interp> and <interpGrp> tags in a similar manner. The interpretation itself can be expressed in an <interpGrp> element, which would replace the <spanGrp> in the example shown above:

```
<interpGrp resp=TMA type='structural unit'>
 <interp id=intro    value='introduction'   >
 <interp id=conflict value='conflict'       >
 <interp id=climax   value='climax'         >
 <interp id=revenge  value='revenge'        >
 <interp id=reconcil value='reconciliation' >
 <interp id=afterm   value='aftermath'      >
</interpGrp>
```

This <interpGrp> element would be linked to the text either by means of the ana attribute, or by means of <link> elements. Using the ana attribute (on <seg> elements introduced specifically for this purpose), the text would be encoded as follows:

```
<p id=P1>
<seg id=S1-S3 ana=intro>
<s id=S1>Sigmund ... was a king in Frankish country.</s>
<s id=S2>Sinfiotli was the eldest of his sons.</s>
<s id=S3>Borghild, Sigmund's wife, had a brother ... </s>
</seg>
<s id=S4a ana=conflict>But Sinfiotli ... wooed the same woman</s>
<s id=S4b ana=climax>and Sinfiotli killed him over it.</s>
<seg id=S5-S17 ana=revenge>
<s id=S5>And when he came home, ... she was obliged to accept it.</s>
<s id=S6>At the funeral feast Borghild was serving beer.</s>
<!-- ... -->
<s id=S17>Sinfiotli drank it off and at once fell dead.</s>
</seg>
<anchor id=nil1 ana=reconcil>
<p id=P2>Sigmund carried him a long way in his arms ... </p>
<p id=P3>King Sigmund lived a long time in Denmark ... </p>
<p id=P4>Sigmund and all his sons were tall ... </p>
<!-- ... -->
<join id=P2-P4 parts='P2 P3 P4' ana=afterm>
```

The linkage may also be accomplished using a <linkGrp> element, whose content is a set of <link> elements which point to each interpretive element and its corresponding text unit. This method does not require the use of the ana attribute on the text units.

```
<linkGrp resp=TMA argTypes='text interpretation' extendArgs=N>
  <link targets='intro    S1-S3'>
  <link targets='conflict S4a'>
  <link targets='climax   S4b'>
  <link targets='revenge  S5-S17'>
  <link targets='reconcil nil1'>
  <link targets='afterm   P2-P4'>
</linkGrp>
```

One obvious advantage of using <interp> rather than elements for the Sigmund text is that the <interp> elements can be reused for marking up other texts in the same document, whereas the elements cannot. Another is that the <interp> element can be used to provide interpretations for discontinuous text elements (represented by <join> elements). On

the other hand, the use of <interp> elements may require the creation of special text elements not otherwise needed (e.g. the <seg> and the <join> in the revised encoding of the text), whereas the the use of elements does not.

The formal declarations for the , <spanGrp>, <interp> and <interpGrp> elements are:

```
<!-- 15.3:  Spans                                              -->
<!ELEMENT span          - O  EMPTY                              >
<!ATTLIST span               %a.global;
                             %a.interpret;
          value              CDATA              #REQUIRED
          from               IDREF              #REQUIRED
          to                 IDREF              #IMPLIED       >
<!ELEMENT spanGrp       - -  ((span)*)                          >
<!ATTLIST spanGrp            %a.global;
                             %a.interpret;                      >
<!ELEMENT interp        - O  EMPTY                              >
<!ATTLIST interp             %a.global;
                             %a.interpret;
          value              CDATA              #REQUIRED      >
<!ELEMENT interpGrp     - -  ((interp)*)                        >
<!ATTLIST interpGrp          %a.global;
                             %a.interpret;                      >
<!-- This fragment is used in sec. 15                          -->
```

15.4 Linguistic Annotation

By *linguistic annotation* we mean here any annotation determined by an analysis of linguistic features of the text, excluding as borderline cases both the formal structural properties of the text (e.g. its division into chapters or paragraphs) and descriptive information about its context (the circumstances of its production, its genre or medium). The structural properties of any TEI-conformant text should be represented using the structural elements discussed elsewhere in this chapter and in chapters 6 ('Elements Available in All TEI Documents') on p. 119, 7 ('Default Text Structure') on p. 183, and the various chapters of Part III (on base tag sets). The contextual properties of a TEI text are fully documented in the TEI Header, which is discussed in chapter 5 ('The TEI Header') on p. 77, and in section 23.2 ('Contextual Information') on p. 540.

Other forms of linguistic annotation may be applied at a number of levels in a text. A code (such as a word-class or part-of-speech code) may be associated with each word or token, or with groups of such tokens, which may be continuous, discontinuous or nested. A code may also be associated with relationships (such as cohesion) perceived as existing between distinct parts of a text. The codes themselves may stand for discrete non-decomposable categories, or they may represent highly articulated bundles of textual features. Their function may be to place the annotated part of the text somewhere within a narrowly linguistic or discoursal domain of analysis, or within a more general semantic field, or any combination drawn from these and other domains.

The manner by which such annotations are generated and attached to the text may be entirely automatic, entirely manual or a mixture. The ease and accuracy with which analysis may be automated may vary with the level at which the annotation is attached. The method employed should be documented in the <interpretation> element within the encoding description of the TEI Header, as described in section 5.3.3 ('The Editorial Practices Declaration') on p. 96. Where different parts of a language corpus have used different annotation methods, the decls attribute may be used to indicate the fact, as further discussed in section 23.3 ('Associating Contextual Information with a Text') on p. 550.

As one example of such types of analysis, consider the following sentence, taken from the Lancaster/IBM Treebank Project.[3]

[3] See G. N. Leech and R. G. Garside, *Running a Grammar Factory,* in *English Computer Corpora: Selected Papers and Research*

"The victim's friends told police that Kruger drove into the quarry and never surfaced."

Our discussion focuses on the way that this sentence might be analysed using the Claws system developed at the University of Lancaster, but exactly the same principles may be applied to a wide variety of other systems.[4]

Output from the system consists of a segmented and tokenized version of the text, in which word class codes have been associated with each token. For our example sentence, we might conveniently represent these codes using entity references:[5]

```
<s>The&AT victim&NN1;'s&GEN friends&NN2 told&VVD police&NN2
that&CST Krueger&NP1  drove&VVD into&II the&AT
quarry&NN1  and&CC  never&RR surfaced&VVD;.&PUN
</s>
```

The names used for these entity references have some significance for the human reader (AT for *article*, NN1 for *singular noun*, NN2 for *plural noun*, etc.), but their representation in the output from an SGML system processing the document may be adjusted, by varying the entity declarations, to suit the convenience of whatever analytic software is to be used. For example, if the SGML parser operating on this sentence uses a set of entity declarations in the following form, then the word class tags will simply disappear from the output.

```
<!ENTITY AT "">
<!ENTITY NN1 "">
<!ENTITY GEN "">
<!-- ... -->
```

Alternatively, suppose the entity set in use follows the following pattern:

```
<!ENTITY AT  "[definite article]">
<!ENTITY NN1 "[singular noun]">
<!ENTITY GEN "[genitive suffix]">
<!-- ... -->
```

Then the sample sentence will be processed by an SGML-aware processor as if it began:

```
The[definite article] victim[singular noun]'s[genitive suffix] ...
```

It would be more useful if the replacement texts for each entity were a code of some significance to a particular analysis program. If the codes are considered to be *atomic*, then one of the mechanisms based on the <interp> element described in section 15.3 ('Spans and Interpretations') on p. 387 is sufficient. If the codes are considered to be compositional (for example that NN1 and NN2 have something in common, namely their *noun-ness*, which they do not share with, say, VVD), then this compositionality may be most clearly expressed using a mechanism based on the <fs> element defined in chapter 16 ('Feature Structures') on p. 397. For a detailed example, see 16.10 ('Two Illustrations') on p. 427.

One such replacement for the word-class entity references above is a set of empty <ptr> elements bearing target attributes as described in section 6.6 ('Simple Links and Cross References') on p. 147. The required entity definitions would look as follows.

```
<!ENTITY   AT   "<ptr target=AT>" >
<!ENTITY   NN1  "<ptr target=NN1>">
<!ENTITY   GEN  "<ptr target=GEN>">
<!-- ... -->
```

Then the text would be expanded to read:

Guide, ed. S. Johansson and A.-B. Stenstrøm (Berlin: de Gruyter; New York: Mouton, 1991), pp. 15-32.

This sentence and its analysis are reproduced by kind permission of the University of Lancaster's Unit for Computer Research on the English Language.

[4]For the word-class tagging method used by Claws see

I. Marshall, *Choice of Grammatical Word Class without Global Syntactic Analysis: Tagging Words in the LOB Corpus,* in *Computers and the Humanities* 17 (1983): 139-50..

For an overview of the system see

R. G. Garside, G. N. Leech, and G. R. Sampson, *The Computational Analysis of English: a Corpus-Based Approach* (Oxford: Oxford University Press, 1991).

[5]We have replaced the Claws code $ for the 's' morpheme by GEN, as in the tag set used by the British National Corpus (see 16.10 ('Two Illustrations') on p. 427), and the code . for the final full stop by PUN.

```
<s>The<ptr target=AT> victim<ptr target=NN1>'s<ptr target=GEN>
friends<ptr target=NN2> told<ptr target=VVD> police
<ptr target=NN2> that<ptr target=CST> Krueger<ptr target=NP1>
drove<ptr target=VVD> into<ptr target=II> the<ptr target=AT>
quarry<ptr target=NN1 and<ptr target=CC> never<ptr target=RR>
surfaced<ptr target=VVD>.<ptr target=PUN>
</s>
```

The <ptr> elements are designed to point to elements with unique identifiers. But we have yet to specify what those elements are. Suppose we say that they are <interp> elements whose values are the same as their identifiers. That is, we provide an <interpGrp> element as follows:

```
<interpGrp type='word classes'>
  <interp id=AT  value=AT>
  <interp id=NN1 value=NN1>
  <interp id=GEN value=GEN>
  <!-- ... -->
</interpGrp>
```

Although common practice, this (or any similar) method of relating text to interpretation is seriously flawed. The interpretations are related not to text elements, but to points in the text, namely those that are occupied by the <ptr> elements. In order to relate the interpretation to the appropriate text units, a uniform convention needs to be applied; for example, that an interpretation relates to all the text material preceding the <ptr> element that points to it up to the immediately preceding <ptr>, or up to the <s> that delimits the S-unit containing that <ptr> element, whichever is nearer. While this convention works with texts that are marked up solely with <ptr> elements that point to interpretation elements, it does not work with texts with additional markup, for example <ptr> elements that are used for some other purpose. In addition, the convention fails for any markup in which interpretations are intended to be associated with nested text elements.

None of these difficulties arise if the text is fully segmented, using the linguistic segment elements described in section 15.1 ('Linguistic Segment Categories') on p. 382, and the ana attribute to point to the interpretations that are associated with each such segment, as follows:

```
<s type=sentence>
<w ana=AT >The</w>
<w ana=NN1>victim</w>
<m ana=GEN>'s</m>
<w ana=NN2>friends</w>
<w ana=VVD>told</w>
<w ana=NN2>police</w>
<w ana=CST>that</w>
<w ana=NP1>Krueger</w>
<w ana=VVD>drove</w>
<w ana=II >into</w>
<w ana=AT >the</w>
<w ana=NN1>quarry</w>
<w ana=CC >and</w>
<w ana=RR >never</w>
<w ana=VVD>surfaced</w>
<c ana=PUN>.</c>
</s>
```

Analysis into phrase and clause elements can be superimposed on the word and morpheme tagging in the preceding illustration. For example, Claws provides the following constituent analysis of the sample sentence (the word class codes have been deleted):

"[N [G The victim's G] friends N] [V told [N police N] [Fn that [N Krueger N] [V [V& drove [P into [N the quarry N]P]V&] and [V+ never surfaced V+]V]Fn]V]"

Treating the labels on the brackets as phrase or clause interpretations, this analysis of the structure of the example sentence can be combined with the word class analysis and represented as follows (the symbol V& representing the first part of a coordinate phrase, has been replaced by V1, and V+, representing the second part, has been replaced by V2).

```
<s type=sentence>
<phr ana=N>
  <phr ana=G>
    <w ana=AT>The</w>
    <w ana=NN1>victim</w>
    <m ana=GEN>'s</m>
    </phr>
  <w ana=NN2>friends</w>
  </phr>
<phr ana=V>
  <w ana=VVD>told</w>
  <phr ana=N><w ana=NN2>police</w></phr>
  <cl ana=Fn>
    <w ana=CST>that</w>
    <phr ana=N><w ana=NP1>Krueger</w></phr>
    <phr ana=V>
      <phr ana=V1>
        <w ana=VVD>drove</w>
        <phr ana=P>
          <w ana=II>into</w>
          <phr ana=N>
            <w ana=AT>the</w>
            <w ana=NN1>quarry</w>
            </phr>
          </phr>
        </phr>
      <w ana=CC>and</w>
      <phr ana=V2>
        <w ana=RR>never</w>
        <w ana=VVD>surfaced</w>
        </phr>
      </phr>
    </cl>
  </phr>
<c ana=PUN>.</c>
</s>
```

A representation using the <linkGrp> element can be obtained by supplying each linguistic segment with its own id attribute, removing its ana attribute, and putting each segment-interpretation pair into a <link> element inside the <linkGrp> element.

Each linguistic segment so far discussed has been well-behaved with respect to the basic document hierarchy, having only a single parent. Moreover, the segmentation has been complete, in that each part of the text is accounted for by some segment at each level of analysis, without discontinuities or overlap. This state of affairs does not of course apply in all types of analysis, and these Guidelines provide a number of mechanisms to support the representation of discontinuities or multiple analyses. A brief overview of these facilities is provided in chapter 31 ('Multiple Hierarchies') on p. 633; also see 14 ('Linking, Segmentation, and Alignment') on p. 331. These mechanisms all depend to a greater or lesser degree on the ability to associate a unique identifier with any element in a TEI-conformant text, and then to specify that identifier as the target of a pointing element of some kind.

The mechanisms proposed in this chapter may also be used to encode analyses of an entirely different kind, for example discourse function. Here is an application of the span technique to record details of a sales transaction in a spoken text.

```
<u who=P1 id=U1>Can I have ten oranges and a kilo of
    bananas please?</u>
<u who=P2 id=U2>Yes, anything else?</u>
<u who=P1 id=U3>No thanks.</u>
<u who=P2 id=U4>That'll be dollar forty.</u>
<u who=P1 id=U5>Two dollars</u>
<u who=P1 id=U6>Sixty, eighty, two dollars. Thank you.</u>
```

```
<spanGrp type=transactions>
  <span from=U1 value='sale request'>
  <span from=U2 to=U3 value='sale compliance'>
  <span from=U4 value='sale'>
  <span from=U5 value='purchase'>
  <span from=U6 value='purchase closure'>
  </spanGrp>
```

For further discussion of the <u> (utterance) element and other elements recommended for transcriptions of spoken language, see chapter 11 ('Transcriptions of Speech') on p. 249.

Chapter 16

Feature Structures

16.1 Introduction

A *feature structure* is a general purpose data structure, which identifies and groups together individual *features*, each of which associates a name with one or more values. Because of the generality of feature structures, they can be used to represent many different kinds of information, and interrelations among various pieces of information, and their instantiation in SGML in these guidelines provides a *metalanguage* for representing text analysis and interpretation. Moreover, this instantiation allows feature values to be of various *types*, and for restrictions to be placed on the values for particular features, by means of *feature system declarations*, which are discussed in chapter 26 ('Feature System Declaration') on p. 589. These restrictions provide the basis for at least partial validation of the feature-structure encodings that are used.

This chapter is organized as follows. Following this introduction, section 16.2 ('Elementary Feature Structures: Features with Binary Values') on p. 398 introduces the *binary* feature values, and shows how elementary feature structures using features with those values may be constructed. Section 16.3 ('Feature, Feature-Structure and Feature-Value Libraries') on p. 400 introduces the tags that represent *libraries* of features, feature structures and feature values, along with methods for pointing at features, feature structures and feature values in these libraries. Section 16.4 ('Symbolic, Numeric, Measurement, Rate and String Values') on p. 402, presents the tags for *symbolic*, *numeric*, *measurement*, *rate*, and *string* values. Section 16.5 ('Structured Values') on p. 408, shows how to use feature-structures themselves as values, thus enabling feature structures to be recursively defined. Section 16.6 ('Singleton, Set, Bag and List Collections of Values') on p. 410 demonstrates the use of multiple values for features, for encoding *set*, *bag*, and *list* collections of values. Section 16.7 ('Alternative Features and Feature Values') on p. 413 presents various methods for representing alternations (disjunctions) of features and feature values. Section 16.8 ('Boolean, Default and Uncertain Values') on p. 417, presents tags for *boolean*, *default*, and *uncertain* values, along with methods for *underspecifying* feature values. Section 16.9 ('Indirect Specification of Values Using the rel Attribute') on p. 421 shows how to specify various logical relations, such as negation and subsumption, between the expressed values for a feature and its actual values. Finally, section 16.10 ('Two Illustrations') on p. 427, illustrates how feature structures may be linked to to text elements.

This tag set is selected as described in 3.3 ('Invocation of the TEI DTD') on p. 39; in a document which uses the markup described in this chapter, the document type declaration should contain the following declaration of the entity *TEI.fs*, or an equivalent one:

```
<!ENTITY % TEI.fs 'INCLUDE'>
```

The entire document type declaration for a document using this additional tag set together with the base tag set for prose might look like this:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
   <!ENTITY TEI.prose     'INCLUDE' >
   <!ENTITY TEI.fs  'INCLUDE' >
```

```
]>
```

The overall document type declaration for this additional tag set has the following structure:

```
<!-- 16.1:  Feature Structures                                  -->
<!-- Text Encoding Initiative: Guidelines for Electronic       -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.     -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy     -->
<!-- in any form is granted, provided this notice is           -->
<!-- included in all copies.                                   -->

<!-- These materials may not be altered; modifications to      -->
<!-- these DTDs should be performed as specified in the        -->
<!-- Guidelines in chapter "Modifying the TEI DTD."            -->

<!-- These materials subject to revision. Current versions     -->
<!-- are available from the Text Encoding Initiative.          -->

<!-- ... declarations from section 16.2                        -->
<!--     (Feature structures, binary values)                   -->
<!--     go here ...                                           -->
<!-- ... declarations from section 16.3                        -->
<!--     (Feature libraries)                                   -->
<!--     go here ...                                           -->
<!-- ... declarations from section 16.4                        -->
<!--     (Symbolic, etc. values)                               -->
<!--     go here ...                                           -->
<!-- ... declarations from section 16.6                        -->
<!--     (Null values)                                         -->
<!--     go here ...                                           -->
<!-- ... declarations from section 16.7                        -->
<!--     (Alternative features and feature values)             -->
<!--     go here ...                                           -->
<!-- ... declarations from section 16.8                        -->
<!--     (Boolean, default, uncertainty values)                -->
<!--     go here ...                                           -->
```

16.2 Elementary Feature Structures: Features with Binary Values

The fundamental elements of a feature structure system are <f> (for *feature*) and <fs> (for *feature structure*). The <fs> element has a type attribute for indicating what type of feature structure it represents, and may contain any number of <f> elements. An <f> element, in turn, has a required name attribute and any number of associated *values*. Feature values may be binary, numeric, symbolic (i.e. taken from a restricted set of legal values), or string-valued, or may consist of sets, lists, or bags of binary, numeric, symbolic, or string values. Specialized values may also be given which allow partial underspecification of the feature. These possible types of values are described in more detail in this and the following sections.

This section considers the special case of feature structures that contain features whose single value is one of the *binary values* represented by the empty elements <plus> and <minus>. The elements which are used for representing feature structures, features and the binary values, along with their descriptions and attributes, are the following.

<fs> analyzes a collection of features and feature alternations as a structural unit. Attributes include:

type provides a type for a feature structure.
feats pointer to features.

rel indicates the relation of the given content to the actual content or value of the feature structure. Legal values are:
- ***eq*** indicates that the actual content is that given.
- ***ne*** indicates that the actual content is not that given.
- ***sb*** indicates that the actual content is subsumed by the given content.
- ***ns*** indicates that the actual content is not subsumed by the given content.

<f> associates a name with a value of any of several different types. Attributes include:

name provides a name for a feature.

org indicates organization of given value or values as singleton, set, bag or list. Legal values are:
- ***single*** indicates that the given value is a singleton.
- ***set*** indicates that the given values are organized as a set.
- ***bag*** indicates that the given values are organized as a bag (multiset).
- ***list*** indicates that the given values are organized as a list.

fVal points to the id attributes of feature values.

rel indicates the relation between the values that are given as the content of the feature or pointed at by the fVal attribute and the actual values of the feature. Legal values are:
- ***eq*** indicates that the given values are the actual values.
- ***ne*** indicates that the given values are not the actual values.
- ***sb*** indicates that the given values are a subset, subbag or sublist of the actual values.
- ***ns*** indicates that the given values are not a subset, subbag or sublist of the actual values.

<plus> provides binary plus value for a feature.

<minus> provides binary minus value for a feature.

The attributes not discussed in this section are discussed in following sections as follows: the feats and the fVal attributes in section 16.3 ('Feature, Feature-Structure and Feature-Value Libraries') on p. 400, the rel attribute in section 16.9 ('Indirect Specification of Values Using the rel Attribute') on p. 421, and the org attribute in section 16.6 ('Singleton, Set, Bag and List Collections of Values') on p. 410.

An <fs> element containing <f> elements with binary values can be straightforwardly used to encode the *matrices* of feature-value specifications for phonetic segments, such as the following for the English segment [s].

```
+---           ---+
| + consonantal   |
| - vocalic       |
| - voiced        |
| + anterior      |
| + coronal       |
| + continuant    |
| + strident      |
+---           ---+
```

Using the additional tag set for feature structures, this might be encoded as follows. Note that <fs> elements may have a type attribute indicating the kind of feature structure in question.

```
<fs type='phonological segment'>
  <f name=consonantal><plus></f>
  <f name=vocalic>    <minus></f>
  <f name=voiced>     <minus></f>
  <f name=anterior>   <plus></f>
  <f name=coronal>    <plus></f>
  <f name=continuant> <plus></f>
  <f name=strident>   <plus></f>
  </fs>
```

The restriction of specific features to specific types of values (e.g. the restriction of the feature 'strident' to the values <plus> or <minus>) cannot be validated by an SGML parser. To enable an application program to check that only legal values for particular features appear, one may write a *feature-system declaration*; see chapter 26 ('Feature System Declaration') on p. 589.

Here are the formal declarations of the <fs>, <f>, <plus> and <minus> elements.

```
<!-- 16.2:  Feature structures, binary values                    -->
<!ELEMENT fs            - -  ((f | fAlt | alt)*)                  >
<!ATTLIST fs                 %a.global;
          type               CDATA              #IMPLIED
          feats              IDREFS             #IMPLIED
          rel                (eq | ne | sb | ns) sb              >
<!ELEMENT f             - O  (null | (plus | minus | any | none
                             | dft | uncertain | sym | nbr |
                             msr | rate | str | vAlt | alt |
                             fs)*)                               >
<!ATTLIST f                  %a.global;
          name               NMTOKEN            #REQUIRED
          org                (single | set | bag | list)
                                                #IMPLIED
          rel                (eq | ne | sb | ns) eq
          fVal               IDREFS             #IMPLIED        >
<!ELEMENT plus          - O  EMPTY                               >
<!ATTLIST plus               %a.global;                          >
<!ELEMENT minus         - O  EMPTY                               >
<!ATTLIST minus              %a.global;                          >
<!-- This fragment is used in sec. 16.1                         -->
```

16.3 Feature, Feature-Structure and Feature-Value Libraries

As the example in the preceding section illustrates, the direct encoding of features structures can be verbose. Consequently, the effort of encoding large numbers of feature structures in this manner could be enormous, and could result in the creation of enormous files. To reduce the size and complexity of the task of encoding feature structures, one may use the feats attribute of the <fs> element to point to one or more of the features of that element. This indirect method of encoding feature structures presumes that the <f> elements are assigned unique SGML id values, and are collected together in <fLib> elements (*feature libraries*). In turn, feature structures can be collected together in <fsLib> elements (*feature-structure libraries*). Finally, one may use the fVal attribute of the <f> element to point to its values. This indirect method of encoding feature values presumes that the value elements are assigned id specifications, and are collected together in <fvLib> elements (*feature-value libraries*). The elements which are used for representing feature, feature-structure and feature-value libraries, along with their descriptions and attributes, are the following.

<fLib> assembles library of feature elements. Attributes include:

type indicates type of feature library (i.e., what kind of features it contains).

<fsLib> assembles library of feature structure elements. Attributes include:

type indicates type of feature-structure library (i.e., what type of feature structures it contains).

<fvLib> assembles library of feature value elements. Attributes include:

type indicates type of feature-value library (i.e., what type of feature values it contains).

For example, suppose a feature library for phonological feature specifications is set up as follows.

```
<fLib type='phonological features'>
  <f id=cns1 name=consonantal><plus> </f>
  <f id=cns0 name=consonantal><minus></f>
  <f id=voc1 name=vocalic>    <plus> </f>
  <f id=voc0 name=vocalic>    <minus></f>
  <f id=voi1 name=voiced>     <plus> </f>
  <f id=voi0 name=voiced>     <minus></f>
```

```
    <f id=ant1 name=anterior>   <plus> </f>
    <f id=ant0 name=anterior>   <minus></f>
    <f id=cor1 name=coronal>    <plus> </f>
    <f id=cor0 name=coronal>    <minus></f>
    <f id=cnt1 name=continuant> <plus> </f>
    <f id=cnt0 name=continuant> <minus></f>
    <f id=str1 name=strident>   <plus> </f>
    <f id=str0 name=strident>   <minus></f>
  <!-- ... -->
    </fLib>
```

Then the feature structures that represent the analysis of the phonological segments (phonemes) /t/, /d/, /s/ and /z/ can be defined as follows.

```
<fs feats='cns1 voc0 voi0 ant1 cor1 cnt0 str0'></fs>
<fs feats='cns1 voc0 voi1 ant1 cor1 cnt0 str0'></fs>
<fs feats='cns1 voc0 voi0 ant1 cor1 cnt1 str1'></fs>
<fs feats='cns1 voc0 voi1 ant1 cor1 cnt1 str1'></fs>
```

The preceding are but four of the 128 logically possible fully specified phonological segments using the seven binary features listed in the feature library. Presumably not all combinations of features correspond to phonological segments (there are no strident vowels, for example). The legal combinations, however, can be collected together in a *feature-structure library*, with each element being given a unique id attribute, as in the following example.

```
<fsLib id=fsl1 type='phonological segment definitions'>
<!-- ... -->
  <fs id=t.df feats='cns1 voc0 voi0 ant1 cor1 cnt0 str0'></fs>
  <fs id=d.df feats='cns1 voc0 voi1 ant1 cor1 cnt0 str0'></fs>
  <fs id=s.df feats='cns1 voc0 voi0 ant1 cor1 cnt1 str1'></fs>
  <fs id=z.df feats='cns1 voc0 voi1 ant1 cor1 cnt1 str1'></fs>
<!-- ... -->
  </fsLib>
```

Text elements can be linked to these feature structures in any of the ways described in section 15.2 ('Global Attributes for Simple Analyses') on p. 387. In the following example, a <linkGrp> element is used to link selected characters in the text 'Caesar seized control' to their phonological representations.

```
<text id=txt1> <!-- ... -->
<body> <!-- ... -->
<s id=s1>
  <w id=s1w1>
    <c id=s1w1c1>C</c>ae<c id=s1w1c2>s</c>ar</w>
  <w id=s1w2>
    <c id=s1w2c1>s</c>ei<c id=s1w2c2>z</c>e<c id=s1w2c3>d</c></w>
  <w id=s1w3>
    con<c id=s1w3c1>t</c>rol</w>.
  </s>
<!-- ... -->
</body></text>

<fsLib id=fsl1 type='phonological segment definitions'>
<!-- as in previous example -->
  </fsLib>

<linkGrp type='phonological identification of characters'
    domains='fsl1 txt1'
    targFunc='phonological.segment character'
    extendArgs=repeat>
<!-- ... -->
  <link id=lt targets='s.df s1w3c1'>
  <link id=ld targets='z.df s1w2c3'>
  <link id=ls targets='s.df s1w1c1 s1w2c1'>
```

```
  <link id=lz targets='z.df s1w1c2 s1w2c2'>
<!-- ... -->
  </linkGrp>
```

Because of the simplicity of the binary feature values, there is no particular gain in pointing at those values rather than specifying them directly. However, the mechanism of using the fVal attribute on <f> elements is useful for representing more complex feature values, and can be illustrated using binary values. Suppose the <plus> and <minus> elements are collected together in a <fvLib>, as follows.

```
<fvLib type='binary values'>
  <plus  id=b1>
  <minus id=b0>
  </fvLib>
```

Then the feature library presented at the beginning of this section can be represented as follows.

```
<fLib type='phonological features'>
  <f id=cns1 name=consonantal fVal=b1></f>
  <f id=cns0 name=consonantal fVal=b0></f>
  <f id=voc1 name=vocalic     fVal=b1></f>
  <f id=voc0 name=vocalic     fVal=b0></f>
  <f id=voi1 name=voiced      fVal=b1></f>
  <f id=voi0 name=voiced      fVal=b0></f>
  <f id=ant1 name=anterior    fVal=b1></f>
  <f id=ant0 name=anterior    fVal=b0></f>
  <f id=cor1 name=coronal     fVal=b1></f>
  <f id=cor0 name=coronal     fVal=b0></f>
  <f id=cnt1 name=continuant  fVal=b1></f>
  <f id=cnt0 name=continuant  fVal=b0></f>
  <f id=str1 name=strident    fVal=b1></f>
  <f id=str0 name=strident    fVal=b0></f>
<!-- ... -->
  </fLib>
```

Although <fs> elements are legitimate feature values (see section 16.5 ('Structured Values') on p. 408), they are not allowed within <fvLib> elements. They should be placed in <fsLib> elements.

Here are the formal declarations of the <fLib>, <fsLib> and <fvLib> elements.

```
<!-- 16.3:  Feature libraries                                    -->
<!ELEMENT fLib          - -  ((f | fAlt)*)                       >
<!ATTLIST fLib               %a.global;
          type               CDATA              #IMPLIED      >
<!ELEMENT fsLib         - -  ((fs | vAlt)*)                      >
<!ATTLIST fsLib              %a.global;
          type               CDATA              #IMPLIED      >
<!ELEMENT fvLib         - -  ((plus | minus | any | none | dft
                             | uncertain | null | sym | nbr |
                             msr | rate | str | vAlt)*)        >
<!ATTLIST fvLib              %a.global;
          type               CDATA              #IMPLIED      >
<!-- This fragment is used in sec. 16.1                         -->
```

16.4 Symbolic, Numeric, Measurement, Rate and String Values

By separating out feature values as content of <f> elements, we are able to classify those values into *types*. In section 16.2 ('Elementary Feature Structures: Features with Binary Values') on p. 398, the two empty elements which represent binary values are defined. In this section,

we define five more feature-value elements: the empty elements **<sym>** for expressing *symbolic values*, **<nbr>** for expressing *numeric values*, **<msr>** for expressing *measurement values*, and **<rate>** for expressing *rate values*; and the element **<str>** for expressing *string values*. These elements, along with their descriptions and attributes, are the following.

<sym> provides symbolic values for features. Attributes include:

- **value** provides a symbolic value for a feature, one of a finite list that may be specified in a feature declaration.
- **rel** indicates the relation of the given value to the actual value. Legal values are:
  - ***eq*** indicates that the actual value is that given.
  - ***ne*** indicates that the actual value is not that given.

<nbr> provides a numeric value or range of values for a feature. Attributes include:

- **value** provides a numeric value.
- **valueTo** together with **value** attribute, provides a range of numeric values.
- **type** indicates whether value or range is to be understood as real or integer. Legal values are:
  - ***int*** specifies that value is an integer; if noninteger is given as value of **value**, then only integer part is used.
  - ***real*** specifies that value is a real number.
- **rel** indicates the relation of the given value or range to the actual value or range. Legal values are:
  - ***eq*** indicates that the actual value or range is that given.
  - ***ne*** indicates that the actual value or range is not the value or range given.
  - ***lt*** indicates that the actual value or range is less than the given value or range.
  - ***le*** indicates that the actual value or range is less than or equal to the given value or range.
  - ***gt*** indicates that the actual value or range is greater than the given value or range.
  - ***ge*** indicates that the actual value or range is greater than or equal to the given value or range.

<msr> provides a measure value or range of values for a feature. Attributes include:

- **unit** provides a unit for a measure feature, one of a finite list that may be specified in a feature declaration.
- **value** provides a numeric value.
- **valueTo** together with **value** attribute, provides a range of numeric values.
- **type** indicates whether value or range is to be understood as real or integer. Legal values are:
  - ***int*** specifies that value is an integer; if noninteger is given as value of **value**, then only integer part is used.
  - ***real*** specifies that value is a real number.
- **rel** indicates the relation of the given value or range to the actual value or range. Legal values are:
  - ***eq*** indicates that the actual value or range is that given.
  - ***ne*** indicates that the actual value or range is not the value or range given.
  - ***lt*** indicates that the actual value or range is less than the given value or range.
  - ***le*** indicates that the actual value or range is less than or equal to the given value or range.
  - ***gt*** indicates that the actual value or range is greater than the given value or range.
  - ***ge*** indicates that the actual value or range is greater than or equal to the given value or range.

<rate> provides a rate value or range of values for a feature. Attributes include:

- **unit** provides a unit for a rate feature, one of a finite list that may be specified in a feature declaration.
- **per** provides an interval for a rate feature, one of a finite list that may be specified in a feature declaration.
- **value** provides a numeric value.
- **valueTo** together with **value** attribute, provides a numeric range of values.
- **type** indicates whether value is to be understood as real or integer. Legal values are:

int specifies that value is an integer; if noninteger is given as value of value, then only integer part is used.

real specifies that value is that of a real number.

rel indicates the relation of the given value or range to the actual value or range. Legal values are:

eq indicates that the actual value or range is that given.

ne indicates that the actual value or range is not the value or range given by the element.

lt indicates that the actual value or range is less than the given value or range.

le indicates that the actual value or range is less than or equal to the given value or range.

gt indicates that the actual value or range is greater than the given value or range.

ge indicates that the actual value or range is greater than or equal to the given value or range.

<str> provides a string value for a feature. Attributes include:

rel indicates the relation of the given value to the actual value. Legal values are:

eq indicates that the actual value is that given.

ne indicates that the actual value is not that given.

sb indicates that the value given is a substring of the actual value.

ns indicates that the value given is not a substring of the actual value.

lt indicates that the actual value is less than the given value.

le indicates that the actual value is less than or equal to the given value.

gt indicates that the actual value is greater than the given value.

ge indicates that the actual value is greater than or equal to the given value.

The <sym> element is to be used for the value of a feature when that feature can have any of a small, finite set of possible values, representable as character strings. For example, consider the problem of specifying the grammatical *case*, *gender* and *number* features of classical Greek noun forms. Assuming that the case feature can take on any of the five values nominative, genitive, dative, accusative and vocative; that the gender feature can take on any of the three values feminine, masculine, and neuter; and that the number feature can take on either of the values singular and plural, then the following may be used to represent the claim that noun form 'theás' 'goddesses' has accusative case, feminine gender and plural number.

```
<fs type='word structure'>
<!-- ... -->
  <f name=case><sym value=accusative></f>
  <f name=gender><sym value=feminine></f>
  <f name=number><sym value=plural></f>
<!-- ... -->
  </fs>
```

Note that instead of using a symbolic value for grammatical number, one could have named the feature *singular* or *plural* and given it an appropriate binary value, as in the following example. Whether one uses a binary or symbolic value in situations like this is largely a matter of taste.

```
<fs type='word structure'>
<!-- ... -->
  <f name=case><sym value=accusative></f>
  <f name=gender><sym value=feminine></f>
  <f name=singular><minus></f>
<!-- ... -->
  </fs>
```

An SGML validator by itself cannot determine that particular values do or do not go with particular features; in particular, it cannot distinguish between the presumably legal encodings in the preceding two examples and the presumably illegal encoding in the following example.

```
<!-- *PRESUMABLY ILLEGAL* ... -->
<fs type='word structure'>
<!-- ... -->
  <f name=case><sym value=feminine></f>
```

```
  <f name=gender><sym value=accusative></f>
  <f name=number><minus></f>
<!-- ... -->
  </fs>
```

There are two ways of attempting to ensure that only legal combinations of feature names and values are used. First, if the total number of legal combinations is relatively small, one can simply list all of those combinations in <fLib> elements (together possibly with <fvLib> elements), and point to them using the feats attribute in the enclosing <fs> element. This method is suitable in the situation described above, since it requires specifying a total of only ten (5 + 3 + 2) combinations of features and values. Further, to ensure that the features are themselves combined legally into feature structures, one can put the legal feature structures inside <fsLib> elements. A total of 30 feature structures (5 x 3 x 2) is required to enumerate all the legal combinations of individual case, gender and number values in the preceding illustration. Of course, the legality of the markup requires that the feat attributes actually point at legally defined features, which an SGML validator, by itself, cannot guarantee.

A more general method of attempting to ensure that only legal combinations of feature names and values are used is to provide a feature system declaration which includes a <valRange> element for each feature one uses. Here is a sample <valRange> element for the <f name=case> element described above; for further discussion of the <valRange> element, see chapter 26 ('Feature System Declaration') on p. 589; the <vAlt> element is discussed in section 16.7 ('Alternative Features and Feature Values') on p. 413.

```
<!-- VALRANGE specification for CASE feature -->
<valRange>
  <vAlt>
    <sym value=nominative>
    <sym value=genitive>
    <sym value=dative>
    <sym value=accusative>
    <sym value=vocative>
    </vAlt>
  </valRange>
```

Similarly, to ensure that only legal combinations of features are used as the content of feature structures, one should provide <fsConstraint> elements for each of the types of feature structure one employs. For discussion of the <fDecl> and <fsConstraint> elements, see 26 ('Feature System Declaration') on p. 589. Validation of the feature structures used in a document based on the feature-system declaration, however, requires that there be an application program that can use the information contained in the feature-system declaration.

Features with <sym>, <plus>, and <minus> values may be used to encode highly structured information such as may be obtained from precoded survey instruments. We illustrate by means of a coding scheme based on the one that is used for classifying potential printed entries in the British National Corpus. The scheme uses the following features and associated values.

medium books and magazines; miscellaneous; written to be spoken
domain imaginative; applied science; arts; belief and thought; commerce and finance; leisure; natural and pure science; social science; world affairs
level high; medium; low
sampling range beginning; middle; end; whole; whole less ten percent
date of origination 1960-1975; 1975-1993
published (miscellaneous items only) yes; no
selection method (books and periodicals only) chosen on grounds of circulation or influence; chosen at random

A comprehensive feature library for this scheme is the following; the id specifications are those currently used in the BNC project.

```
<fLib type='BNC classification features'>
  <f id=CA002 name=medium><sym value=book.or.periodical></f>
  <f id=CA003 name=medium><sym value=miscellaneous></f>
```

```
<f id=CA004 name=medium><sym value=written.to.be.spoken></f>
<f id=CA005 name=domain><sym value=imaginative></f>
<f id=CA006 name=domain><sym value=applied.science></f>
<f id=CA007 name=domain><sym value=arts></f>
<f id=CA008 name=domain><sym value=belief.and.thought></f>
<f id=CA009 name=domain><sym value=commerce.and.finance></f>
<f id=CA00A name=domain><sym value=leisure></f>
<f id=CA00B name=domain><sym value=natural.and.pure.science></f>
<f id=CA00C name=domain><sym value=social.science></f>
<f id=CA00D name=domain><sym value=world.affairs></f>
<f id=CA00E name=level><sym value=high></f>
<f id=CA00F name=level><sym value=medium></f>
<f id=CA00G name=level><sym value=low></f>
<f id=CA00H name=sample.type><sym value=beginning></f>
<f id=CA00J name=sample.type><sym value=middle></f>
<f id=CA00K name=sample.type><sym value=end></f>
<f id=CA00L name=sample.type><sym value=whole></f>
<f id=CA00M name=sample.type><sym value=whole.less.ten.percent></f>
<f id=CA00N name=published.between><sym value=1960.1975></f>
<f id=CA00P name=published.between><sym value=1975.1993></f>
<f id=CA00R name=published><plus></f>
<f id=CA00S name=published><minus></f>
<f id=CA00T name=selection.method><sym value=principled></f>
<f id=CA00U name=selection.method><sym value=random></f>
</fLib>
```

An entry which is a book or periodical on world affairs, medium level, sampled from the middle, published between 1975 and 1993, and selected on a principled basis could then be assigned the following feature-structure code; this code could also be placed in a feature-structure library that contains all the possible fully-specified BNC entry classifications. This library would have a total of 1620 (3 x 9 x 3 x 5 x 2 x 2) entries.

```
<fs type='BNC classification for written documents'
    id=CA2DFJPT
    feats='CA002 CA00D CA00F CA00J CA00P CA00T'></fs>
```

The <nbr> element is to be used when the value of a feature is a number or a range of numbers. For example, suppose one wishes to encode information contained in classified advertisements for the sale or rental of real estate, such as the number of bedrooms and bathrooms in a listed property, and its advertised selling or rental price. One way of representing such information is as follows.

```
<fs type='real estate listing'>
<!-- ... -->
  <f name=number.of.bathrooms><nbr value=2></f>
  <f name=number.of.bedrooms><nbr value=3></f>
  <f name=monthly.rent><nbr value=625.00></f>
<!-- ... -->
  </fs>
```

The information that the number of bedrooms is in the range from 3 to 5 and the monthly rent is in the range from 625.00 to 950.00 may be represented as follows, using the optional valueTo attribute.

```
<f name=number.of.bedrooms><nbr value=3 valueTo=5></f>
<f name=monthly.rent><nbr value=625.00 valueTo=950.00></f>
```

The <nbr> (and also the <msr> and <rate> elements defined below) element also may have a type attribute to specify whether the values of the value and valueTo attributes are to be construed as integer or real numbers.

The <msr> element to be used when the value of a feature is a scalar quantity, essentially a combination of a numeric value and a symbolic value for identifying the scale on which the numeric value occurs. For example, real estate listings often provide the area (in square feet or

meters) of a house or apartment and the area (in acres or hectares) of land being sold or rented. One way of representing information about such areas is as follows.

```
<fs type='real estate listing'>
<!-- ... -->
  <f name=interior.area><msr unit=sq.ft value=2000></f>
  <f name=property.area><msr unit=acre value=0.5></f>
<!-- ... -->
  </fs>
```

The value of the <f name=monthly.rent> feature in the two examples above might be more accurately analysed as a measurement rather than as a numeric value, since the amount of the rent in question is to be understood as payable in a particular currency, such as US or Canadian dollars or Italian lire. To make the currency scale explicit, the first example of this feature might be re-encoded as follows.

```
<f name=monthly.rent><msr unit=USD value=625.00></f>
```

The unit and value attributes of the <msr> element are both required. If the unit attribute is not needed (for example, if no confusion would result if the unit attribute is not specified), then the <nbr> element may be used to express the feature value.

The <rate> element is to be used when the value of a feature is a rate. This element has a required per attribute for expressing the interval over which the rate is measured (typically, but not necessarily, a temporal interval), and an optional unit attribute for expressing the scalar unit. For example, one can encode the wage rate of USD $8.25 per hour as follows.

```
<f name=wage.rate><rate value=8.25 unit=USD per=hour></f>
```

Note that the <f name=monthly.rent> element illustrated above can be re-encoded as having a rate value, with a per=month attribute, as follows.

```
<f name=rent><rate value=625.00 unit=USD per=month></f>
```

To encode interest, inflation or tax rates, the unit attribute can be used to indicate that the value attribute is to be understood as a percentage. For example, an interest rate of 8.25% per year can be encoded in either of the following two ways.

```
<f name=interest><rate unit=percent value=8.25 per=year></f>

<f name=interest><rate value=0.0825 per=year></f>
```

Finally, the <str> element is to be used for the value of a feature when that value is a string drawn from a very large or potentially unbounded set of possible strings of characters, so that it would be impractical or impossible to use the <sym> element. These values are expressed not as the values of the value attribute, as in the case of symbolic, numeric, measurement and rate values, but as the content of the <str> element. For example, one may encode the street address of a property in a real estate listing, as follows.

```
<fs type='real estate listing'>
<!-- ... -->
  <f name=address><str>3418 East Third Street</str></f>
<!-- ... -->
  </fs>
```

Here are the formal declarations of the <sym>, <nbr>, <msr>, <rate> and <str> elements.

```
<!-- 16.4:  Symbolic, etc. values                              -->
<!ELEMENT sym           - O  EMPTY                              >
<!ATTLIST sym                %a.global;
          value              CDATA               #REQUIRED
          rel                (eq | ne)           eq              >
<!ELEMENT nbr           - O  EMPTY                              >
<!ATTLIST nbr                %a.global;
          value              CDATA               #REQUIRED
          valueTo            CDATA               #IMPLIED
          rel                (eq | ne | lt | le | gt | ge)
                                                 eq
```

```
          type              (int | real)        #IMPLIED       >
<!ELEMENT msr           - O  EMPTY                              >
<!ATTLIST msr                %a.global;
          value             CDATA               #REQUIRED
          valueTo           CDATA               #IMPLIED
          unit              CDATA               #REQUIRED
          rel               (eq | ne | lt | le | gt | ge)
                                                eq
          type              (int | real)        #IMPLIED       >
<!ELEMENT rate          - O  EMPTY                              >
<!ATTLIST rate               %a.global;
          value             CDATA               #REQUIRED
          valueTo           CDATA               #IMPLIED
          unit              CDATA               #IMPLIED
          per               CDATA               #REQUIRED
          rel               (eq | ne | gt | ge | lt | le)
                                                eq
          type              (int | real)        #IMPLIED       >
<!ELEMENT str           - -  (#PCDATA)                          >
<!ATTLIST str                %a.global;
          rel               (eq | ne | sb | ns | lt | le | gt
                            | ge)               eq             >
<!-- This fragment is used in sec. 16.1                       -->
```

16.5 Structured Values

Features may have *structured values* as well; these values are represented by either the <fs> element, or the fVal attribute on the <f> element, which can point to an <fs> element. Since an <fs> or a pointer to an <fs> is permitted to occur as a value of an <f>, recursion is possible. For example, an <fs> element may contain or point to an <f> element, which may contain or point to an <fs> element, which may contain or point to an <f> element, and so on. To illustrate the use of structured values, consider the following simple model of a personal record, consisting of a person's name, date of birth, place of birth, and sex. Each personal record is a <fs type='personal record'> tag, consisting of the corresponding four features, three of which take structured values, as in the following example.

```
<fs type='personal record'>
  <f name=full.name>
    <fs type='name record'>
      <f name=first.name><str>Kathleen</str></f>
      <f name=middle.name><str>Anne</str></f>
      <f name=surname><str>Barnett</str></f>
      </fs>
    </f>
  <f name=date.of.birth>
    <fs type='date record'>
      <f name=year><nbr value=1968></f>
      <f name=month><nbr value=4></f>
      <f name=day><nbr value=17></f>
      </fs>
    </f>
  <f name=place.of.birth>
    <fs type='place record'>
      <f name=city><str>Austin</str></f>
      <f name=state><sym value=TX></f>
      </fs>
    </f>
  <f name=sex><sym value=female>
```

```
</fs>
```

Now suppose that feature-structure libraries are maintained for name records and place records, and that the name record in the previous example is identified with the attribute id=Nkab027 and the place record is identified with the attribute id=txaustin. [1] Then the preceding example could also be encoded as follows. (An identifier is also provided for the personal record.)

```
<fs id=Pkab027 type='personal record'>
  <f name=full.name fVal=Nkab027></f>
  <f name=date.of.birth>
    <fs type='date record'>
      <f name=year><nbr value=1968></f>
      <f name=month><nbr value=4></f>
      <f name=day><nbr value=17></f>
      </fs>
    </f>
  <f name=place.of.birth fVal=txaustin></f>
  <f name=sex><sym value=female>
  </fs>
```

This representation could be simplified further if a feature library is maintained for the year, month, day and sex features, so that the feats attribute may be used as follows.

```
<fs id=Pkab027 type='personal record' feats='sxf'>
  <f name=full.name fVal=Nkab027></f>
  <f name=date.of.birth>
    <fs type='date record' feats='y1968 m04 d17'></fs>
    </f>
  <f name=place.of.birth fVal=txaustin></f>
  </fs>
```

Next, suppose that a feature-structure library is also maintained for personal records, and that the library also contains records for the parents of the individual identified in the previous example. Suppose that the father is identified as Pmfb009 and the mother as Parn002. Then the personal-record feature structure could be easily augmented to include pointers to the parents, as follows.

```
<fs id=Pkab027 type='personal record' feats='sxf'>
  <f name=full.name fVal=Nkab027></f>
  <f name=date.of.birth>
    <fs type='date record' feats='y1968 m04 d17'></fs>
    </f>
  <f name=place.of.birth fVal=austinTX></f>
  <f name=mother fVal=Parn002>
  <f name=father fVal=Pmfb009>
  </fs>
```

If the personal records identified as Parn002 and Pmfb009 also contain information about the parents of those individuals, then from the present record, one would have access to that individual's grandparents as well.

Assuming that personal records of the sort described in this section are being maintained in association with text files, the records can be linked to those texts in any of the ways described in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331, provided that identifiers are added for appropriate features, as in the following illustration.

```
<text id=Bfile>
<!-- ... -->
<div id=Tkab027 type='birth certificate'>
<p><name type=person id=T1kab027>Kathleen Anne Barnett</name>
was born at <time id=T1t0659>6:59 a.m.</time> on
<date id=T1d680417>April 17, 1968</date> in
<name type=org id=T1setonhsp>Seton Hospital</name> in
```

[1] Feature-structure, rather than feature-value, libraries should be used for housing collections of feature structures.

```
<name type=place id=T1txaustin>Austin</name>
to <seg id=s1>Mr.</seg> and <seg id=s2>Mrs.</seg>
<name type=person id=T1mfb009>Michael F. Barnett</name> of
<name type=place id=T1sansabaTX>San Saba</name>.
<!-- ... -->
<join id=T1arn002 targets='s2 T1mfb009'>
<join id=T2mfb009 targets='s1 T1mfb009'>
<!-- ... -->
</text>

<fsLib id=Prec type='personal records'>
<!-- ... -->
<fs id=Pkab027 type='personal record' feats='sxf'>
  <f name=full.name fVal=Nkab027></f>
  <f id=Dkab027 name=date.of.birth>
    <fs type='date record' feats='y1968 m04 d17'></fs>
    </f>
  <f id=Bkab027 name=place.of.birth fVal=txaustin></f>
  <f id=Mkab027 name=mother fVal=Parn002></f>
  <f id=Fkab027 name=father fVal=Pmfb009></f>
  </fs>
<!-- ... -->
</fsLib>

<linkGrp  type='record verification'
          domains='Bfile Prec'
          targFunc='source goal'
          extendTarg=0>
<!-- ... -->
  <link targets='T1kab027 Nkab027'>
  <link targets='T1d680417 Dkab027'>
  <link targets='T1txaustin Bkab027'>
  <link targets='T1arn002 Mkab027'>
  <link targets='T2mfb009 Fkab027'>
<!-- ... -->
  </linkGrp>
```

16.6 Singleton, Set, Bag and List Collections of Values

In the discussion to this point, we have assumed that features have exactly one simple value. However, for many purposes, it is useful to be able to consider the values of certain features to be organized in more complex ways, for example as sets, bags (or multisets), or lists. Accordingly, we provide for four different ways in which feature values may be organized, namely as *singletons*, *sets*, *bags* and *lists*. We do so by means of an `org` attribute on the `<f>` element, which takes on one of the designated values `single`, `set`, `bag`, and `list`. A feature whose value is organized as a singleton is understood as having exactly one simple value. If more than one value is specified for it, we assume that only the first one is considered to be its true value. A feature whose value is organized as a set, bag or list may have any positive number of values as its content. Sets and bags are distinguished from lists in that the order in which the values are specified does not matter for the former, but does matter for the latter. Sets are distinguished from bags and lists in that repetitions of values do not count for the former but do count for the latter. SGML does not provide a way of validating that values for features organized as sets are not repeated; such validation would have to be carried out by an application program. Our method of representing set, bag and list values also does not permit such values to be directly embedded within one another. In order to embed a set within a set, for example, one must specify the embedded set as

the value of a feature of a feature-structure value of the including set.[2]

No default value for the org attribute is declared in the DTD; however, a default value for that attribute can be declared for particular features in the feature-system declaration; see 26 ('Feature System Declaration') on p. 589. Note that if only one value is specified for a given <f> element, the set, bag and list values of the org are all essentially equivalent to the singleton value, so the omission of the org attribute for such a feature is not problematic.[3]

To illustrate the use of the org attribute, suppose that the illustration of personal records from the previous section is extended to include pointers to an individual's siblings. Suppose also that the individual identified as <fs id=Pkab027> has siblings identified as <fs id=Panb005>, <fs id=Pmfb010> and <fs id=Pzrb001> in the personal records library. Then we may extend the personal record for <fs id=Pkab027> as follows.

```
<fs id=Pkab027 type='personal record' feats='sxf'>
  <f name=full.name fVal=Nkab027></f>
  <f name=date.of.birth>
    <fs type='date record' feats='y1988 m04 d17'></fs>
    </f>
  <f name=place.of.birth fVal=austinTX></f>
  <f name=mother fVal=Parn002>
  <f name=father fVal=Pmfb009>
  <f name=siblings org=set fVal='Panb005 Pmfb010 Pzrb001'>
  </fs>
```

A more elaborate illustration of the use of the org attribute is the the following <f name=career org=list> element which may be added to the personal records of an individual to record the job career of that individual. The feature structures which constitute the value of this feature document the jobs which the individual has held in the order in which they were held. Note that a list has been embedded within a list by means of intervening <fs type='employment record'> and <f name=promotion.history> elements.

```
<f name=career org=list>
  <fs type='employment record'>
    <f name=employer><str>Safeway Stores</str></f>
    <f name=hiring.information>
      <fs type='hire structure'>
        <f name=hire.date>
          <fs type='date structure' feats='y1988 m06'></fs>
          </f>
        <f name=job.title><sym value=stocker></f>
        <f name=wage><rate value=6.00 per=hour></f>
        <f name=hours.worked><rate value=40 per=week></f>
        <f name=status.code fVal=sc4A></f>
        </fs>
      </f>
    <f name=promotion.history org=list>
      <fs type='promotion record'>
        <f name=date>
          <fs type='date structure' feats='y1988 m12'></fs>
            </f>
        <f name=job.title><sym value=cashier></f>
        <f name=wage><rate value=7.00 per=hour></f>
        <f name=hours.worked><rate value=40 per=week></f>
        <f name=status.code fVal=sc4A></f>
        </fs>
      <fs type='promotion record'>
        <f name=date>
          <fs type='date structure' feats='y1990 m02'></fs>
          </f>
        <f name=job.title><sym value=supervisor></f>
```

[2]This is not as hard as it sounds. The embedding of a list within a list is illustrated in the second example below.

[3]Unless the value is the <null> element; see below.

```
      <f name=salary><rate value=18000 per=year></f>
      <f name=status.code fVal=sc3C></f>
      </fs>
    </f>
  <f name=termination.information>
    <fs type='termination structure'>
      <f name=termination.date>
        <fs type='date structure' feats='y1991 m04'></fs>
          </f>
      <f name=status.code fVal=sc3C></f>
      <f name=reason.for.termination><sym value=laid.off></f>
      </fs>
    </f>
  </fs>
<fs type='employment record'>
  <!-- ... -->
  </fs>
<!-- ... -->
</f>
```

The information contained in such features may be linked to textual references in the usual way. The <f name=status.code> feature has been included to show how evaluative or interpretive information can be included along with information gleaned from textual records. The example presumes that the status code values are maintained in a designated <fvLib>.

Features with values organized as sets, bags or lists can sometimes be used instead of features organized as singletons, whose values are individual feature structures. For example, consider the following encoding of the English verb form 'sinks', which contains a <f name=agreement> element whose value is a feature structure which contains <f name=person> and <f name=number> elements with symbolic values.

```
<fs type='word structure'>
<!-- ... -->
  <f name=word.class><sym value=verb></f>
  <f name=tense><sym value=present></f>
  <f name=agreement>
    <fs type='agreement structure'>
      <f name=person><sym value=third></f>
      <f name=number><sym value=singular></f>
      </fs>
    </f>
<!-- ... -->
  </fs>
```

If one does not care about the names of the features contained within the <fs type='agreement structure'> element, the containing <f name=agreement> element can be given an org attribute with the value set, and the contained <fs> element, together with the person and number feature elements it contained, can be eliminated, as follows.

```
<fs type='word structure'>
<!-- ... -->
  <f name=word.class><sym value=verb></f>
  <f name=tense><sym value=present></f>
  <f name=agreement org=set><sym value=third><sym value=singular></f>
<!-- ... -->
  </fs>
```

The encoding in the preceding example presumes that the <fDecl> element for the <f name=agreement> element would look something like the following; for further details, see 26 ('Feature System Declaration') on p. 589.

```
<fDecl name=agreement org=set>
<!-- ... -->
  <valRange>
```

```
      <vAlt>
        <sym value=first>
        <sym value=second>
        <sym value=third>
        </vAlt>
      <vAlt>
        <sym value=singular>
        <sym value=plural>
        </vAlt>
      </valRange>
  <!-- ... -->
    </fDecl>
```

The set, bag or list which has no members is known as the null (or empty) set, bag or list. To refer to it, the <null> element is provided; its description and attributes are as follows.

<null> represents the null set if org=set is specified for the feature of which it is the value; represents the null bag if org=bag is specified for the feature of which it is the value; represents the null list if org=list is specified for the feature of which it is the value; has no interpretation if
org=single is specified for the feature of which it is the value.

So, for example, to indicate that the individual identified above by the <fs id=Pkab027> element has no siblings, we may specify the <f name=siblings> element as follows.

```
<f name=siblings org=set><null></f>
```

The <null> element when used with a feature organized as a singleton is a semantic error; however, its appearance as a value for such a feature cannot be flagged by SGML. The <null> element, when it appears as a feature value, must be the only value.

Here is the formal declarations of the <null> element.

```
<!-- 16.6:  Null values                                    -->
<!ELEMENT null          - O  EMPTY                          >
<!ATTLIST null               %a.global;                     >
<!-- This fragment is used in sec. 16.1                    -->
```

16.7 Alternative Features and Feature Values

In this section, two methods of representing the alternation (ambiguity or uncertainty) of features and feature values are presented. The first of these methods is to be used for nonsystematic or sporadic markup of alternation of individual features or values; it makes use of the special-purpose <fAlt> and <vAlt> elements. The other is to be used for systematic markup of alternation and for the alternation of groups of features or values; it makes use of the general-purpose <alt> element introduced in section 14.8 ('Alternation') on p. 373. The <fAlt> and <vAlt> elements have the following description and attributes.

<fAlt> provides alternative features for a feature structure or other feature alternation. Attributes include:

mutExcl indicates whether values are mutually exclusive. Legal values are:

Y indicates that the values are mutually exclusive.

N indicates that the values are not mutually exclusive.

<vAlt> provides alternative (disjunctive) values for a feature. Attributes include:

mutExcl indicates whether values are mutually exclusive. Legal values are:

Y indicates that the values are mutually exclusive.

N indicates that the values are not mutually exclusive.

To illustrate the use of the <fAlt> element to represent the alternation of features, suppose one is uncertain whether a particular real estate advertisement describes a house with two bedrooms or with two bathrooms. This uncertainty can be represented as follows.

```
<fs type='real estate listing'>
<!-- ... -->
  <fAlt>
    <f name=number.of.bathrooms><nbr value=2></f>
    <f name=number.of.bedrooms><nbr value=2></f>
    </fAlt>
<!-- ... -->
  </fs>
```

This representation leaves unspecified whether or not the alternation is *mutually exclusive* (i.e. whether having two bathrooms excludes the possibility of having two bedrooms and vice versa). To make this aspect of the alternation explicit, one can specify a value for the mutExcl attribute, as follows.

```
<fs type='real estate listing'>
<!-- ... -->
  <fAlt mutExcl=N>
    <f name=number.of.bathrooms><nbr value=2></f>
    <f name=number.of.bedrooms><nbr value=2></f>
    </fAlt>
<!-- ... -->
  </fs>
```

The <fAlt> element can also be used to represent uncertainty about whether the number of bathrooms is two or three, as follows; note that the attribute value mutExcl=Y can be inferred for the <fAlt> element in this example.

```
<fs type='real estate listing'>
<!-- ... -->
  <fAlt>
    <f name=number.of.bathrooms><nbr value=2></f>
    <f name=number.of.bathrooms><nbr value=3></f>
    </fAlt>
<!-- ... -->
  </fs>
```

However, the <f name=number.of.bathrooms> element in this example can be factored out of the alternation, and a <vAlt> element used instead to represent the alternation of just the feature values, as follows.

```
<fs type='real estate listing'>
<!-- ... -->
  <f name=number.of.bathrooms>
    <vAlt><nbr value=2><nbr value=3></vAlt>
    </f>
<!-- ... -->
  </fs>
```

The <fAlt> and <vAlt> elements can also be used to indicate certain alternations among values of features organized as sets, bags or lists. For example, suppose one uses a <f name=extras org=set> element in feature structures for real estate listings to represent items that are mentioned to enhance a property's sales value, such as whether it has a pool or a good view. Now suppose for a particular listing, the extras include an alarm system and a fenced-in yard, and either a pool or a jacuzzi (but not both). This situation could be represented, using the <vAlt> element, as follows.

```
<fs type='real estate listing'>
<!-- ... -->
  <f name=extras org=set>
    <str>alarm system</str>
    <str>fenced-in yard</str>
    <vAlt mutExcl=Y>
      <str>pool</str>
      <str>jacuzzi</str>
      </vAlt>
```

```
    </f>
<!-- ... -->
  </fs>
```

Now suppose the situation is like the preceding except that one is also uncertain whether the property has an alarm system or a fenced-in yard, or possibly both. This can be represented as follows.

```
<fs type='real estate listing'>
<!-- ... -->
  <f name=extras org=set>
    <vAlt mutExcl=N>
      <str>alarm system</str>
      <str>fenced-in yard</str>
      </vAlt>
    <vAlt mutExcl=Y>
      <str>pool</str>
      <str>jacuzzi</str>
      </vAlt>
    </f>
<!-- ... -->
  </fs>
```

Finally, suppose that the listing specifies that the property has a finished basement, and that it also has either an alarm system and a pool or a fenced-in yard and a jacuzzi. This situation cannot be represented using the <vAlt> element, because the alternation holds between subsets of two values each. It can, however, be represented using the <fAlt> element, as follows; note that the <str> element with the value `finished basement` element must be repeated.

```
<fs type='real estate listing'>
<!-- ... -->
  <fAlt mutExcl=Y>
    <f name=extras org=set>
      <str>finished basement</str>
      <str>alarm system</str>
      <str>pool</str>
      </f>
    <f name=extras org=set>
      <str>finished basement</str>
      <str>fenced-in yard</str>
      <str>jacuzzi</str>
      </f>
    </fAlt>
<!-- ... -->
  </fs>
```

If a large number of ambiguities or uncertainties involving a relatively small number of features and values need to be represented, it is recommended that the general-purpose <alt> element discussed in section 14.8 ('Alternation') on p. 373 be used, rather than the special-purpose <fAlt> and <vAlt> elements. The use of the <alt> element avoids the need to explictly represent the alternating elements more than once.

For example, suppose one has set up a <fsLib> element containing feature structures representing the morphological structures of classical Greek inflected words, along with collections of individual features and feature values, encoded by <fLib> and <fvLib> elements as appropriate. The following example shows how one might then represent the morphological structure of a feminine gender, accusative case, plural number noun form in classical Greek, such as 'theás' 'goddesses' discussed in section 16.4 ('Symbolic, Numeric, Measurement, Rate and String Values') on p. 402:

```
<fsLib type='noun structures'>
  <!-- ... -->
  <!-- plural accusative feminine noun -->
  <fs id=WnGfKaNp type='noun structure' feats='Wn Gf Ka Np'></fs>
```

```
  <!-- ... -->
  </fsLib>

<fLib type='morphological features'>
  <f id=Wn  name=word.class fVal=nn></f>
  <!-- ... -->
  <f id=Gf  name=gender fVal=fe></f>
  <!-- ... -->
  <f id=Ka  name=case fVal=ac></f>
  <!-- ... -->
  <f id=Np  name=number fVal=pl></f>
  <!-- ... -->
  </fLib>

<fvLib type='morphological feature values'>
  <!-- ... -->
  <sym id=nn value=noun>
  <!-- ... -->
  <sym id=fe value=feminine>
  <!-- ... -->
  <sym id=ac value=accusative>
  <!-- ... -->
  <sym id=pl value=plural>
  <!-- ... -->
  </fvLib>
```

Now consider the noun form 'theaí' 'goddesses', which is analyzable as a feminine plural noun form in either the nominative or the vocative case. We may represent this ambiguity by adding the following entries to the <fsLib>, <fLib>, and <fvLib> elements in the preceding example; assume that appropriate entries for unambiguous nominative and vocative case forms have already been entered.

```
<!-- Add the following to the feature-structure library (FSLIB) -->
  <!-- plural nominative-or-vocative feminine noun -->
  <fs id=WnGfKnvNp type='noun structure' feats='Wn Gf Knv Np'></fs>

<!-- Add the following to the feature library (FLIB) -->
  <!-- CASE=nominative or vocative -->
  <f id=Knv name=case fVal=novo></f>

<!-- Add the following to the feature value library (FVLIB) -->
  <!-- nominative or vocative -->
  <alt id=novo targets='no vo'>
```

If the <fvLib> element is not used, and specifications for particular feature values are entered as content of the <f name=...> elements in the <fLib> element, then the ambiguity can be represented as follows.

```
<fsLib type='noun structures'>
  <!-- ... -->
  <!-- plural nominative-or-vocative feminine noun -->
  <fs id=WnGfKnvNp type='noun structure' feats='Wn Gf Knv Np'></fs>
  <!-- ... -->
  </fsLib>

<fLib type='morphological features'>
  <!-- ... -->
  <f id=Kn name=case><sym value=nominative>
  <!-- ... -->
  <f id=Kv name=case><sym value=vocative>
  <!-- ... -->
  <alt id=Knv targets='Kn Kv'>
```

```
  <!-- ... -->
  </fLib>
```

The <alt> element together with the <join> element can, unlike the <fAlt> and <vAlt> elements, be used to express alternations between sets of features. An example of such an alternation is found in certain feminine gender Greek noun forms ending in '-as', such as 'peíras' 'attempt(s)', which may be analyzed as having either genitive case and singular number features or accusative case and plural number features, as follows (again, assuming the existence of other elements and identifier attributes for simple features and values).

```
<!-- Add the following to the FSLIB element -->
  <!-- feminine noun, either genitive singular or accusative plural -->
  <fs id=WnGfKg.NsKa.Np type='noun structure' feats='Wn Gf Kg.NsKa.Np'>
    </fs>

<!-- Add the following to the FLIB element  -->
  <join id=Kg.Ns targets='Kg Ns'> <!-- genitive singular -->
  <join id=Ka.Np targets='Ka Np'> <!-- accusative plural -->

  <!-- alternation:  gen. sg. or acc. plural -->
  <alt id=Kg.NsKa.Np targets='Kg.Ns KaNp'>
```

Here are the formal declarations of the <fAlt> and <vAlt> elements.

```
<!-- 16.7:  Alternative features and feature values          -->
<!ELEMENT fAlt          - -  ((f | fs | fAlt), (f | fs |
                              fAlt)+)                        >
<!ATTLIST fAlt                %a.global;
          mutExcl             (Y | N)             #IMPLIED   >
<!ELEMENT vAlt          - -  ((plus | minus | any | none | dft
                              | uncertain | null | sym | nbr |
                              msr | rate | str | vAlt | fs),
                              (plus | minus | any | none | dft |
                              uncertain | null | sym | nbr | msr
                              | rate | str | vAlt | fs)+)    >
<!ATTLIST vAlt                %a.global;
          mutExcl             (Y | N)             #IMPLIED   >
<!-- This fragment is used in sec. 16.1                       -->
```

16.8 Boolean, Default and Uncertain Values

In this section we define four special empty elements used as feature values: the *boolean value* elements <any> and <none>, the <dft> element, and the <uncertain> element.

The boolean value elements are used to indicate whether the features they are associated with have values. The element <any> corresponds to the boolean value `true` (i.e., that the feature it is associated with has a value — not the same as the binary value `plus`), and the element <none> corresponds to the boolean value `false` (i.e., that the feature it is associated with has no value). The <dft> element is used to indicate that the feature it is associated with has its default value in the feature structure in which it appears. Finally, the <uncertain> element may be used to indicate uncertainty about what value, if any, its associated feature has; it is equivalent to the alternation of the <any> and <none> elements. To indicate uncertainty about which of the possible legal values a particular feature has, one should use the <any> element.

The descriptions and attributes of these elements are as follows.

<any> represents boolean *true* value variable.
<none> represents boolean *false* value variable.
<dft> provides default value for a feature.
<uncertain> provides uncertainty value for a feature.

The values <null> and <none> are distinct. The former is to be used with a feature organized as a set, bag, or list to indicate that its value is the null set, bag, or list in a particular

feature structure. The latter is to be used with such a feature to indicate that it has no value in a particular feature structure.

The *boolean* values <any> and <none> are also distinct from the *binary* values <plus> and <minus>. The latter pair are specific possible values for features, whereas the former pair represent *ranges* of possible values, not specific possible values, for features. For example, suppose that the <valRange> element for the <f name=auxiliary> element is declared as follows in the feature structure declaration, so that either boolean value is legal.

```
<!-- VALRANGE tag for the AUXILIARY feature -->
<valRange><vAlt><plus><minus></vAlt></valRange>
```

Then the following two pairs of specifications are distinct.

```
<f name=auxiliary><plus></f>  =/= <f name=auxiliary><any></f>

<f name=auxiliary><minus></f> =/= <f name=auxiliary><none></f>
```

In this situation, the <any> element is equivalent to the alternation of the <plus> and <minus> elements, and the <none> element is equivalent to the negation of that alternation.

However, if the auxiliary feature is declared to take only the <plus> value, then the first pair of the specifications below are equivalent, but the second is not; in fact, the first member of the second pair is invalid.

```
<f name=auxiliary><plus></f>  ==  <f name=auxiliary><any></f>

<f name=auxiliary><minus></f> =/= <f name=auxiliary><none></f>
```

It is even possible to declare that a particular feature can never have values, as follows for the feature <f name=impossible>.

```
<!-- VALRANGE tag for the IMPOSSIBLE feature -->
<valRange></valRange>
```

In this case, the following specifications are equivalent.

```
<f name=impossible><any></f>  == <f name=impossible><none></f>
```

The elements <any> and <dft> are also designed to be used in conjunction with the <fDecl> and <valDefault> elements in the feature system declaration discussed in section 26 ('Feature System Declaration') on p. 589. First, consider the <any> element, and suppose that the <valRange> element in the <fDecl> element for the <f name=gender> element is specified as follows.

```
<!-- VALRANGE for the GENDER feature -->
<valRange>
  <vAlt>
    <sym value=feminine>
    <sym value=masculine>
    <sym value=neuter>
    </vAlt>
  </valRange>
```

Then the following two representations are equivalent.

```
<f name=gender><any></f>

<f name=gender>
  <vAlt>
    <sym value=feminine>
    <sym value=masculine>
    <sym value=neuter>
    </vAlt>
  </f>
```

Second, consider the <dft> element, and suppose that the default value for the <f name=gender> element is specified in the <valDefault> element of its <fDecl> element as having the value <sym value=feminine>. Then the following three representations are equivalent; note that if a <f name=...> element appears without content and without a valid fVal attribute, then it is equivalent to the same element with the <dft> element as its content.

```
<f name=gender></f>

<f name=gender><dft></f>

<f name=gender><sym value=feminine></f>
```

Using the <any> and <dft> elements, together with an <fDecl> element for the corresponding feature in the feature system declaration, provides a method for *underspecifying* the value of that feature. The <any> element means that the associated feature has a legal value but what value it has is not specified. The <dft> element means that the associated feature has the value which the encoder has declared is the normal value of the feature.

The boolean elements <any> and <none> also have specific uses within <fsConstraints> and <fDecl> elements in feature system declarations, as described in chapter 26 ('Feature System Declaration') on p. 589. For example, the element <any> can appear as the value of a feature contained within an <fs> of a particular type which appears in the <cond> element of an <fsConstraints> element, to indicate that the feature must appear in feature structures of the designated type (i.e., that it is obligatory) and that when it does appear, it may appear with any of its legal values. Similarly, <none> can appear in this way to specify that the feature cannot be present in feature structures of the indicated type (i.e., that it is obligatorily absent from such feature structures). All other features that are declared to have values are understood to be optional in such feature structures.

For example, the following may appear as part of the <fsConstraints> of a feature system declaration to indicate that a <fs type='agreement structure'> must contain a legal instance of the <f name=number> element but must not contain a legal instance of the <f name=category> element.

```
<cond><fs type='agreement structure'></fs>
<then><fs>
         <f name=number><any></f>
         <f name=category><none></f>
         </fs>
```

Further constraints can be imposed on a feature structure of a particular type in the <valRange> elements of features which take feature structures of that type as values. For example, suppose that verb and adjective agreement in German are represented by feature structures of the following sorts, in which verb forms agree in person and number with their subjects and adjective forms agree in gender, case, and number with their subjects.

```
<fs type='verb structure'>
<!-- ... -->
  <f name=verbAgreement>
    <fs type='agreement structure'>
      <f name=person><sym value=first></f>
      <f name=number><sym value=plural></f>
      </fs>
    </f>
<!-- ... -->
  </fs>

<fs type='adjective structure'>
<!-- ... -->
  <f name=adjAgreement>
    <fs type='agreement structure'>
      <f name=gender><sym value=feminine></f>
      <f name=case><sym value=accusative></f>
      <f name=number><sym value=plural></f>
      </fs>
    </f>
<!-- ... -->
  </fs>
```

In order to ensure that a <fs type='agreement structure'> tag which appears as the

value of a <f name=verbAgreement> element may be specified for any person and number feature, but for no gender and case feature, we may provide a <valRange> element for the verbAgreement feature as follows.

```
<!--VALRANGE specification for the VERBAGREEMENT feature -->
<valRange>
  <fs type='agreement structure'>
    <f name=person><any></f>
    <f name=case><none></f>
    <f name=gender><none></f>
    <f name=number><any></f>
    </fs>
  </valRange>
```

Similarly, to ensure that a <fs type='agreement structure'> element which appears as the value of a <f name=adjAgreement> element may be specified for any case, gender, and number feature, but for no person feature, we may provide a <valRange> element for the adjAgreement feature as follows.

```
<valRange>
  <fs type='agreement structure'>
    <f name=person><none>
    <f name=case><any>
    <f name=gender><any>
    <f name=number><any>
    </fs>
  </valRange>
```

The combination of declarations like these and the principle of *subsumption* discussed in 16.9 ('Indirect Specification of Values Using the rel Attribute') on p. 421, allows feature structures to be underspecified in text markup. For example, to indicate that a given adjective inflection feature (tagged <f name=adjInflection>) is a feature structure (tagged <fs type='inflection structure'>) specifying plural number and any gender and case, we can omit the elements for gender and case on the <fs> element, as follows.

```
<f name=adjInflection>
  <fs type='inflection structure'>
    <f name=number><sym value=plural></f>
    </fs>
  </f>
```

When supplied as the value of a verb inflection feature (tagged <f name=verbInflection>), the same feature structure would be interpreted as an inflection structure specifying plural number and any person.

If an optional feature is not specified in a feature-structure value, then it is assumed to occur with the <uncertain> value. For further discussion, see section 16.9 ('Indirect Specification of Values Using the rel Attribute') on p. 421.

Here are the formal declarations of the <any>, <none>, <dft>, and <uncertain> elements.

```
<!-- 16.8:  Boolean, default, uncertainty values              -->
<!ELEMENT any           - O  EMPTY                             >
<!ATTLIST any                %a.global;                        >
<!ELEMENT none          - O  EMPTY                             >
<!ATTLIST none               %a.global;                        >
<!ELEMENT dft           - O  EMPTY                             >
<!ATTLIST dft                %a.global;                        >
<!ELEMENT uncertain     - O  EMPTY                             >
<!ATTLIST uncertain          %a.global;                        >
<!-- This fragment is used in sec. 16.1                       -->
```

16.9 Indirect Specification of Values Using the rel Attribute

The rel attribute is provided for the feature value elements <sym>, <nbr>, <msr>, <rate>, <str>, <fs>, and <default> (but not <plus>, <minus>, <null>, <vAlt>, <any>, <none>, and <uncertain>). This attribute may be used for specifying which of various logical relations the given value has to the actual value of the feature. For all value elements for which the rel attribute is defined, except for <fs>, the default value for that attribute is eq, which means that the actual value is equal (or identical) to the given value. Accordingly, the following representations are both interpreted to mean that the value of the <f name=case> element is the <sym value=genitive> element.

```
<f name=case><sym value=genitive></f>

<f name=case><sym rel=eq value=genitive></f>
```

16.9.1 The Not-Equals Relation

The rel attribute can also be specified as having the value ne, which means that the associated feature has a value which is not equal to the given value. For example, the value <nbr rel=ne value=1> in the following example denotes any legal numeric value for the element <f name=number.of.bathrooms> other than 1.

```
<f name=number.of.bathrooms><nbr rel=ne value=1></f>
```

If an <fDecl> element has been provided which defines the legal values for the associated feature, then the value ne can be given a positive interpretation. For example, suppose that the <valRange> element is declared in the <fDecl> element for the <f name=case> element as follows.

```
<!-- VALRANGE specification in FDECL for CASE feature -->
<valRange>
  <vAlt>
    <sym value=nominative>
    <sym value=genitive>
    <sym value=dative>
    <sym value=accusative>
    <sym value=vocative>
    </vAlt>
  </valRange>
```

Suppose also that the <f name=case> element is declared as obligatory in a particular feature structure. Then the following specifications are equivalent in that structure.

```
<f name=case><sym rel=ne value=genitive></f>

<f name=case>
  <vAlt>
    <sym value=nominative>
    <sym value=dative>
    <sym value=accusative>
    <sym value=vocative>
    </vAlt>
  </f>
```

That is, when the rel attribute occurs with the value ne in the value of an obligatory feature in a feature structure, the actual value of that feature may be any of its legal values *other than* the specified value.

On the other hand, if the <f name=case> feature is declared as optional in a particular feature structure, then the following specifications are equivalent in that structure.

```
<f name=case><sym rel=ne value=genitive></f>

<f name=case>
  <vAlt>
```

```
    <sym value=nominative>
    <sym value=dative>
    <sym value=accusative>
    <sym value=vocative>
    <none>
    </vAlt>
  </f>
```

That is, when the rel attribute has the value ne in the value of an optional feature in a feature structure, the actual value of that feature may be any of its legal values other than the specified value, or <none>.

If the rel attribute is specified with the value ne for a <nbr>, <msr>, or <rate> element for which the valueTo attribute is also specified, then the actual range may be any range distinct from that given. For example, the following means that the number of bathrooms is a range distinct from 3 to 5 (e.g., 3 to 4, 3 to 6, 4 to 5, 4 to 6, 0 to 2, etc.).

```
<f name=number.of.bathrooms><nbr rel=ne value=3 valueTo=5></f>
```

16.9.2 Other Inequality Relations

For the elements <nbr>, <msr>, <rate>, and <str>, the rel attribute may also take on the following values; the use of these values for the <str> element presumes that a particular character and string ordering (or *sorting*) convention is understood.

lt The actual value or range is any legal value or range less than the specified value or range.

le The actual value or range is any legal value or range less than or equal to the specified value or range.

gt The actual value or range is any legal value or range greater than the specified value or range.

ge The actual value or range is any legal value or range greater than or equal to the specified value or range.

These attribute values may be used as shown in the following examples. The first states that the number of bedrooms is less than 5; the second that an illegal speed is any speed greater than 65 miles per hour; the third that a lot size is in a range which is less than or equal to the range of from 5 to 10 acres;[4]

the fourth that the last name is any string greater than the empty string (i.e., any nonempty string, given normal string-ordering conventions); and the fifth that for a feature whose value is a list of two strings, the first precedes the string 'M' and the second is the string 'M', or any string following it.

```
<f name=number.of.bedrooms><nbr rel=lt value=5></f>

<f name=illegal.speed><rate rel=gt value=65 unit=miles per=hour></f>

<f name=lot.size><msr rel=le value=5 valueTo=10 unit=acre></f>

<f name=last.name><str rel=gt></str></f>

<f name=pairs org=list><str rel=lt>M</str><str rel=ge>M</str></f>
```

16.9.3 Subsumption and Non-subsumption Relations

When the rel attribute is given the values sb or ns, the markup expresses the claim that the value given subsumes, or does not subsume, the actual value for the feature in question.

On the <str> element, these values are used to specify that the string value given in the <str> element is or is not a *substring* of the actual value of the feature. The first example below specifies that the actual feature value may be any string at all (since the empty string is a substring of every

[4] We say that one range is less than or equal to another if both the value and valueTo attributes of the first are less than or equal to the corresponding attributes of the second.

string), the second that it might be any string in which the string 'the' occurs as a substring, and the third that it might be any string in which the string 'the' does not occur as a substring.

```
<str rel=sb></str>
<str rel=sb>the</str>
<str rel=ns>the</str>
```

On the <fs> element, the attribute values sb and ns indicate that the given feature structure does or does not legally *subsume* the actual feature structure. By definition, one feature structure subsumes another if the second feature structure is identical to the first or contains more information than the first. The default value for the rel attribute of the <fs> element is sb. The subsumption of feature structures is illustrated by the following four examples; suppose that the <f name=person> and <f name=number> elements are either optional or obligatory in these <fs type='agreement structure'> example elements.

```
<fs id=p3ns type='agreement structure'>
  <f name=person><sym value=third></f>
  <f name=number><sym value=singular></f>
  </fs> <!-- 3d person singular -->

<fs id=p3nx type='agreement structure'>
  <f name=person><sym value=third></f>
  </fs> <!-- 3d person -->

<fs id=pxns type='agreement structure'>
  <f name=number><sym value=singular></f>
  </fs> <!-- singular -->

<fs id=pxnx type='agreement structure'>
  </fs> <!-- -->
```

The fourth example, "pxnx", subsumes all four of the examples, since each contains at least as much information as does feature structure "pxnx". Conversely, the first example, "p3ns", subsumes only itself. Finally, the second and third examples, identified as "p3nx" and "pxns" attributes, subsume themselves and the first feature structure, but not each other.

If both person and number are obligatory features of agreement structure elements, then the last three elements in the preceding list have the same interpretation as their counterparts in the following list.

```
<fs id=p3na type='agreement structure'>
  <f name=person><sym value=third></f>
  <f name=number><any></f>
  </fs> <!-- 3d person -->

<fs id=pans type='agreement structure'>
  <f name=person><any></f>
  <f name=number><sym value=singular></f>
  </fs> <!-- singular -->

<fs id=pana type='agreement structure'>
  <f name=person><any></f>
  <f name=number><any></f>
  </fs> <!-- -->
```

On the other hand, if both person and number are optional features of agreement structures, then those three elements have the same interpretation as their counterparts in the following list.

```
<fs id=p3nu type='agreement structure'>
  <f name=person><sym value=third></f>
  <f name=number><uncertain></f>
  </fs> <!-- 3d person -->

<fs id=puns type='agreement structure'>
  <f name=person><uncertain></f>
```

```
  <f name=number><sym value=singular></f>
  </fs> <!-- singular -->

<fs id=punu type='agreement structure'>
  <f name=person><uncertain></f>
  <f name=number><uncertain></f>
  </fs> <!-- -->
```

That is, if an optional feature is omitted from a feature-structure representation, then that feature may have any of its legal values or the value <uncertain>.

The value sb is chosen as the default value for the rel attribute of the <fs> element, because it provides for the most economical means for underspecifying them. One situation in which it may be preferable to specify <fs rel=eq> is when the feature structure has many optional features and it is known that none of them occur.

The specification <fs rel=ns> is used to denote the feature structures that the specified feature structure does not subsume. This provides a handy way of saying that a certain combination of features is not present, for example the combination of third person and singular number, as in the agreement structure of the English verb form 'sink', understood as a present tense verb form. The following example expresses the claim that third-person and singular-number features are not both present in the agreement feature, but makes no further claim about what is present.

```
<f name=agreement>
  <fs id=Np3ns rel=ns type='agreement structure'>
    <f name=person><sym value=third></f>
    <f name=number><sym value=singular></f>
    </fs>
  </f>
```

In most real situations, of course, one can infer, from the range of possible values for person and number, what the remaining possibilities are. Suppose, for example, that in the relevant feature system declaration, the features 'person' and 'number' are given the following <valRange> elements:

```
<!-- VALRANGE for the PERSON feature  -->
<valRange>
  <vAlt>
    <sym value=first>
    <sym value=second>
    <sym value=third>
    </vAlt>
  </valRange>

<!-- VALRANGE for the NUMBER feature  -->
<valRange>
  <vAlt>
    <sym value=singular>
    <sym value=plural>
    </vAlt>
  </valRange>
```

Suppose, further, that the person and number features are obligatory in feature structures of the type "agreement structure". Then the element <fs id=Np3ns> above is equivalent to the following alternation; the features whose value is <any> may be omitted, since they are implied by the default value of sb for the rel attribute in the enclosing <fs> elements.

```
<vAlt id=p12na-panp>
  <fs id=p12na type='agreement structure'>
    <f name=person>
      <vAlt><sym value=first><sym value=second></vAlt></f>
    <f name=number><any></f>
    </fs>
  <fs id=panp type='agreement structure'>
```

```
    <f name=person><any></f>
    <f name=number><sym value=plural></f>
    </fs>
  </vAlt>
```

If, on the other hand, the person and number features were optional in feature structures of type "agreement structure", then the interpretation of an underspecified feature structure will change. The element <fs id=Np3ns> given above is then equivalent to the following alternation; the features whose value is <uncertain> may be omitted as they are implied by the default subsumption relation holding between the structure given and the actual structure.

```
<vAlt id=p120nu-punp0>
  <fs id=p120nu type='agreement structure'>
    <f name=person>
      <vAlt><sym value=first><sym value=second><none></vAlt>
      </f>
    <f name=number><uncertain>
    </fs>
  <fs id=punp0 type='agreement structure'>
    <f name=person><uncertain>
    <f name=number>
      <vAlt><sym value=plural><none></vAlt>
      </f>
    </fs>
  </vAlt>
```

16.9.4 Relations Holding with Sets, Bags, and Lists

The rel attribute is also provided for the <f> element, but is designed to be used with that element only when its org attribute (see section 16.6 ('Singleton, Set, Bag and List Collections of Values') on p. 410) is set to set, bag, or list. When associated with the <f> element, the rel attribute may take on any of the following four values: eq, ne, sb, and ns. The default value is eq. Consider first the use of the rel attribute with the <f> element when the given value of the feature is <null>.

```
<f name=extras org=set><null></f>

<f name=extras org=set rel=ne><null></f>

<f name=extras org=set rel=sb><null></f>

<f name=extras org=set rel=ns><null></f>
```

The first example states that the 'extras' feature has the null set as its value. The second example states that the 'extras' feature is a set which is not equal to the null set. That is, its actual value might be any non-null set. The third example states that the 'extras' feature has as its value a set of which the null set is a subset; that is to say, any set at all, including the null set. Note that this is not equivalent to the following, which states that the extras feature has as its value a single element which is any legal value for the 'extras' feature, including for example a <str> element containing the value pool.

```
<f name=extras org=set><any></f>
```

Finally, the fourth example states that the 'extras' feature has as its value a set of which the null set is not a subset. Since the null set is a subset of every set, the fourth example in effect claims that the 'extras' feature has no legal value; it is thus equivalent to the following, which states directly that the 'extras' feature has no value.

```
<f name=extras org=set><none></f>
```

Consider next the use of the rel attribute with the <f> element when the given value of the feature is a single <str> element with the content pool:

```
<f name=extras org=set><str>pool</str></f>
```

```
<f name=extras org=set rel=ne><str>pool</str></f>

<f name=extras org=set rel=sb><str>pool</str></f>

<f name=extras org=set rel=ns><str>pool</str></f>
```

The first example states that the value of the 'extras' feature is a set consisting of a single member, namely a <str> element containing the value pool. The second example states that the 'extras' feature has as its value a set which is not equal to the set consisting of this particular member. It could, however, be a two-membered set, one of whose members is some other value. This example is thus not equivalent to the following, which states that the 'extras' feature has as its value a set comprising a single member other than a <str> element with the content pool:

```
<f name=extras org=set><str rel=ne>pool</str></f>
```

The third example states that the 'extras' feature has as its value any set of which the set consisting of the single member specified is a subset (i.e., any set which contains the element <str> with the value pool, and possibly others). Finally, the fourth example states that the 'extras' feature has as its value any set which does not contain this element as a member.

16.9.5 Varieties of Subsumption and Non-subsumption

The rel values sb and ns have different meanings depending on whether they occur within a <str>, <fs> or <f> element. However, the use of a common name for the value reflects a fundamental similarity in those meanings. For example, the value sb can be used in all three elements to indicate that the actual value is any string, any feature structure, or any set, bag or list, as follows. In the second example below, the rel attribute has not been specified, since it has the value sb by default on <fs> elements.

```
<str rel=sb></str>

<fs></fs>

<f name=... org=set rel=sb><null></f>

<f name=... org=bag rel=sb><null></f>

<f name=... org=list rel=sb><null></f>
```

Because the value sb is not defined for the attribute rel on the <nbr>, <msr> and <rate> elements, the indication that a value may be any number, measure or rate is sometimes not quite as simple. Here is one way of specifying any positive or negative integer numeric value.[5]

```
<vAlt>
  <nbr type=int rel=gt value=0>
  <nbr type=int rel=le value=0>
  </vAlt>
```

The value ns also is understood in similar ways in the different elements in which it may occur. Above in this section, the equivalence of the following representations under certain conditions was shown (the id attributes and the redundant features with <any> values have been omitted).

```
<f name=agreement>
  <fs rel=ns type='agreement structure'>
    <f name=person><sym value=third></f>
    <f name=number><sym value=singular></f>
    </fs>
  </f>

<f name=agreement>
```

[5] Typically, there will be no need to use an encoding like this one as the value of a feature, since the <any> element is available for that purpose. However, in setting up the feature declaration for that feature, it may be necessary to use such an encoding, precisely so as to provide an interpretation for the use of the <any> element as the value of that feature.

```
  <vAlt>
    <fs type='agreement structure'>
      <f name=person>
        <vAlt><sym value=first><sym value=second></vAlt>
        </f>
      </fs>
    <fs type='agreement structure'>
      <f name=number><sym value=plural></f>
      </fs>
    </vAlt>
  </f>
```

The value `ns` has an analogous meaning when the value in question is a set rather than a feature structure. Recast in such terms, the equivalence above still holds good:

```
<f name=agreement org=set rel=ns>
  <sym value=third><sym value=singular>
  </f>

<f name=agreement org=set rel=sb>
  <vAlt>
    <vAlt><sym value=first><sym value=second></vAlt>
    <sym value=plural>
    </vAlt>
  </f>
```

16.10 Two Illustrations

In this section, we present two illustrations based on one text of how to associate feature structures and their components with textual elements. Our example text is the article *Memoirs of a Dog Shrink* that appeared in the popular magazine *Dogs Today* in August 1991. This text has been selected for inclusion in the British National Corpus. The first illustration associates the text with a structure that represents a significant portion of the information contained in the text. The second marks up the grammatical structure of the orthographic words and certain other comparable units in the text. Here is the text, with markup provided down to the level of `<s>` elements. The `n` attribute values are taken from the BNC markup; the `id` attribute values have been added for purposes of these illustrations.

```
<div1 id=DT91mds type='article'>
<head rend=italic>
<s id=mds01 n=00732>Memoirs of a Dog Shrink</s></head>
 <head type=sub rend=italic>
<s id=mds04 n=00735>Cartoonist Russell Jones
takes a ramble through Peter Neville's files</s></head>
<list>
<item><s id=mds05 n=00736>Case number: 72</s></item>
<item><s id=mds06 n=00737>Name: Jessie</s></item>
<item><s id=mds07 n=00738>Breed: Collie</s></item>
<item><s id=mds08 n=00739>Problem: Light bulb phobia</s></item>
</list>

<p><s id=mds09 n=00740>Jess the collie was a laid-back
sort of hound who spent most of his life stretched out
on a fireside rug in his large Surrey home.</s></p>
<p><s id=mds10 n=00741>The closest he came to exercise
was to open one eye every so often, if someone entered
the room, or to open both eyes, smile, and wag his
tail as he'd done on one occasion when confronted by a
housebreaker!</s></p>
<p><s id=mds11 n=00742>This extremely lazy lifestyle
```

```
was one long yawn from dawn to dusk.</s>
<s id=mds12 n=00743>Only the odd bouts of involuntary
twitching in his sleep reassured his owner that Jess
was still safe and sound in the land of the
living!</s></p>
<p><s id=mds13 n=00744>One winter night, as the mutt
twitched away in front of the fire, his mind somewhere
between Basingstoke and the twilight zone, a 100-watt
light bulb in the standard lamp above his head
suddenly exploded without warning!</s></p>
<p><s id=mds14 n=00745>According to his owner, who
witnessed the spectacle, Jessie rose gracefully toward
the ceiling like a space shuttle and, after lingering
in mid-air for what seemed an eternity, crashed to the
floor and fled the house with a speed and agility the
owner found quite amazing.</s></p>
<p><s id=mds15 n=00746>Jessie did not return home for
several hours.</s>
<s id=mds16 n=00747>When he eventually did show up, it
was obvious to all that he was a changed dog!</s>
<s id=mds17 n=00748>What plodded through the front door
was not the lovable, lazy hound who had once lived
there but a grim-faced light bulb serial
killer!</s></p>
<p><s id=mds18 n=00749>Within seconds of his return,
Jessie launched a vicious attack on a table lamp,
popping the bulb and wrecking the shade before
charging into the lounge.</s>
<s id=mds19 n=00750>There, in a frenzy of violence, he
reduced the standard lamp to a table lamp in 10
seconds flat!</s></p>
<p><s id=mds20 n=00751>After a room-to-room chase
lasting several minutes, during which every lamp in
the house was turned to sawdust, the dog was finally
caught and wrestled to the ground.</s></p>
<p><s id=mds21 n=00752>With his house plunged into
darkness, Jessie's owner sought my help.</s></p>

<div2 type='part'><head>
<s id=mds22 n=00753>SIMPLE SOLUTION</s></head>
<p><s id=mds23 n=00754>When I first saw the dog, it
was quite obvious he'd been deeply affected by the
explosion and had developed a 100-watt phobia for
light bulbs!</s></p>
<p><s id=mds24 n=00755>By placing his feeding bowl
closer each day to a table lamp the dog gradually
learned to live with his enemy.</s>
<s id=mds25 n=00756>Within a couple of weeks, his
killer instincts had disappeared and he was back where
he belonged &mdash; twitching away peacefully on the
fireside rug.</s></p>
</div2>
</div1>
```

The first illustration is based on the observation that from the example text, it is possible to infer a fairly extensive medical history for the dog described in it. Suppose that we have a definition of a feature structure that represents a canine medical history. Then we can fill in feature values in that history from the text, and prepare a <linkGrp> element that specifies the links between the text segments and the various features specified in the feature structure. Here is a hypothetical example of such a filled-in feature structure and associated link group.

```
<fs type='canine medical history'id=j37>
  <f name=name id=j37pn><str>Jessie</str></f>
  <f name=called.by org=set id=j37pc>
    <str>Jessie</str>
    <str>Jess</str>
    </f>
  <f name=breed id=j37b><sym value=collie>
  <f name=owner id=j37o>
    <fs type='owner description'>
      <f name=name><uncertain></f>
      <f name=address id=j37or><str>Surrey</str></f>
      </fs>
  <f name=illness org=list id=j37i>
    <fs type='case history' id=j37i1>
      <f name=name.of.specialist id=j37i1sn>
        <fs type='name structure'>
          <f name=last.name><str>Neville</str></f>
          <f name=first.name><str>Peter</str></f>
          </fs>
      <f name=title.of.specialist><uncertain>
      <f name=case.number id=j37i1n><nbr value=72></f>
      <f name=age.at.incidence><uncertain></f>
      <f name=date.of.incidence><uncertain>
      <f name=baseline.condition org=set id=j37i1b>
        <sym value=lazy>
        <sym value=friendly>
        <sym value=indoor>
        </f>
      <f name=symptoms id=j37i1s>
        <fs type='symptom structure'>
          <f name=behaviors org=set id=j37i1sb>
            <sym value=agitated>
            <sym value=destructive>
            <sym value=unfriendly>
            </f>
          <f name=particulars id=j37i1sp>
             <str>ran off, then returned and
             destroyed every lamp in the house</str></f>
          </fs>
        </f>
      <f name=diagnosis id=j37i1d>
        <fs type='diagnosis structure'>
          <f name=date.of.diagnosis><uncertain>
          <f name=disease id=j37i1dd>
            <str>light bulb phobia</str>
            </f>
          <f name=presumed.cause id=j37i1dc>
            <str>explosion of light bulb over patient's head</str>
            </f>
          </fs>
        </f>
      <f name=treatment id=j37i1t>
        <fs type='treatment history'>
          <f name=medicine><none></f>
          <f name=regime id=j37i1tr><str>positive reinforcement</str></f>
          <f name=particulars id=j37i1tp>
            <str>systematically decreased distance between
             feeding bowl and table lamp</str></f>
          <f name=duration.of.treatment id=j37i1td>
            <msr unit=week value=2>
```

```
            </f>
          </fs>
        </f>
      <f name=result id=j37i1r>
        <str>return to baseline condition</str>
        </f>
      </fs>
    </f>
  </fs>
<linkGrp domains='j37 DT91mds'
         targFunc='feature segment'
         extendTarg=2>
  <link targets='j37pn    mds06'>
  <link targets='j37pc    mds09 mds11 mds14 mds18 mds21'>
  <link targets='j37b     mds07 mds09'>
  <link targets='j37or    mds09'>
  <link targets='j37i1sn  mds04'>
  <link targets='j37i1n   mds05'>
  <link targets='j37i1b   mds09 mds10 mds11'>
  <link targets='j37i1sb  mds14 mds16 mds17 mds18 mds19 msb20'>
  <link targets='j37i1sp  mds15 mds20'>
  <link targets='j37i1dd  mds08 mds23'>
  <link targets='j37i1dc  mds23'>
  <link targets='j37i1tr  mds24'>
  <link targets='j37i1td  mds25'>
  <link targets='j37r     mds25'>
  </linkGrp>
```

From this illustration, we see that links can be made not only between text and feature structure elements, but also between text and feature elements. For that matter, links between text and feature value elements can also be made.

The second illustration takes advantage of the fact that this text, like others that appear in the BNC, has been provided with detailed grammatical markup of most of its orthographic words and certain other comparable structural units. For example, the second paragraph of the above text has been marked up as follows.

```
<s n=00741>The&AT0; closest&AJS; he&PNP; came&VVD; to&PRP; exercise&NN1;
was&VBD; to&TO0; open&VVI; one&CRD; eye&NN1; every so often&AV0;,&PUN;
if&CJS; someone&PNI; entered&VVD; the&AT0; room&NN1;,&PUN; or&CJC;
to&TO0; open&VVI; both&DT0; eyes&NN2;,&PUN; smile&VVI;,&PUN; and&CJC;
wag&VVI; his&DPS; tail&NN1; as&CJS; he&PNP;'d&VHD; done&VDN; on&PRP;
one&CRD; occasion&NN1; when&AVQ; confronted&VVN; by&PRP; a&AT0;
housebreaker&NN1;!&PUN;</s>
```

The entities that appear in this fragment may be expanded into pointers to feature structures that represent grammatical structure by means of entity definitions as follows.

```
<!-- ... -->
<!ENTITY   AJS    "<ptr target=AJS>"                                  >
<!-- ... -->
<!ENTITY   AT0    "<ptr target=AT0>"                                  >
<!-- ... -->
```

This method of associating feature structures with textual elements has a number of drawbacks, most important of which is the fact that the association is implicit, relying on the relative position of pointer and associated text, rather than being explicit. A better method would be to segment the text into the units under analysis, and point to the feature structures from within the unit tags, by means of the ana attribute (see sections 15.2 ('Global Attributes for Simple Analyses') on p. 387 and 15.4 ('Linguistic Annotation') on p. 392).

```
<s n=00741>
<w ana=AT0>The</w>
<w ana=AJS>closest</w>
```

```
<w ana=PNP>he</w>
<w ana=VVD>came</w>
<w ana=PRP>to</w>
<w ana=NN1>exercise</w>
<w ana=VBD>was</w>
<w ana=TO0>to</w>
<w ana=VVI>open</w>
<w ana=CRD>one</w>
<w ana=NN1>eye</w>
<phr ana=AV0>
  <w>every</w>
  <w>so</w>
  <w>often</w>
  </phr>
<c ana=PUN>,</c>
<w ana=CJS>if</w>
<w ana=PNI>someone</w>
<w ana=VVD>entered</w>
<w ana=AT0>the</w>
<w ana=NN1>room</w>

<!-- ... -->

</s>
```

To provide pointers in both direction between text and structural analysis, one may supply both the text segments and the feature-structure tags with identifiers, and associate the segments with their analysis by means of a <linkGrp> (see section 14.1 ('Pointers') on p. 333), as follows.

First, we define a feature-structure library to represent all of the grammatical structures that are used in the BNC encoding scheme. (For illustrative purposes, we cite here only the structures needed for the first six words of the sample sentence):

```
<fsLib id=BNCgs type='BNC grammatical structures'>
   <!-- ... -->
  <fs type='grammatical structure' id=AJS feats='Wj Ds'></fs>
  <fs type='grammatical structure' id=AT0 feats='Wl'></fs>
  <fs type='grammatical structure' id=PNP feats='Wr Rp'></fs>
  <fs type='grammatical structure' id=VVD feats='Wv Bv Fd'></fs>
  <fs type='grammatical structure' id=PRP feats='Wp Bp'></fs>
  <fs type='grammatical structure' id=NN1 feats='Wn Tc Ns'></fs>
  <!-- ... -->
  </fsLib>
```

It will be noted that each feature structure in this library bears an identifier corresponding with the code supplied as the value for the ana attribute in the sample sentence. The component features of each feature structure are further specified by the feats attribute. These identify one or more <f> elements in the following feature library (again, only a few of the available features are quoted here):

```
<fLib type='BNC grammatical features'>
<!-- ... -->
  <f name=verbBase  id=Bv><sym value=main></f>
  <f name=prepBase  id=Bp><sym value=lexical></f>
  <f name=degree    id=Ds><sym value=superlative></f>
  <f name=verbForm  id=Fd><sym value=ed></f>
  <f name=number    id=Ns><sym value=singular></f>
  <f name=pronType  id=Rp><sym value=personal></f>
  <f name=nounType  id=Tc><sym value=common></f>
  <f name=class     id=Wj><sym value=adjective></f>
  <f name=class     id=Wl><sym value=article></f>
  <f name=class     id=Wn><sym value=noun></f>
```

```
  <f name=class     id=Wp><sym value=preposition></f>
  <f name=class     id=Wr><sym value=pronoun></f>
  <f name=class     id=Wv><sym value=verb></f>
<!-- ... -->
  </fLib>
```

Next, here is a markup of the start of our sample sentence being analyzed, with identifiers for each segment; see section 15.1 ('Linguistic Segment Categories') on p. 382 for discussion of the <phr>, <w>, <m> and <c> elements used here.

```
<s n=00741 id=mds09>
<w id=mds0901>The</w>
<w id=mds0902>closest</w>
<w id=mds0903>he</w>
<w id=mds0904>came</w>
<w id=mds0905>to</w>
<w id=mds0906>exercise</w>
<w id=mds0907>was</w>
<w id=mds0908>to</w>
<w id=mds0909>open</w>
<w id=mds0910>one</w>
<w id=mds0911>eye</w>
<phr id=mds0912>
  <w>every</w>
  <w>so</w>
  <w>often</w>
  </phr>
<c id=mds0913>,</c>
<w id=mds0914>if</w>
<w id=mds0915>someone</w>
<w id=mds0916>entered</w>
<w id=mds0917>the</w>
<w id=mds0918>room</w>
<c id=mds0919>,</c>
<w id=mds0920>or</w>
<w id=mds0921>to</w>
<w id=mds0922>open</w>
<w id=mds0923>both</w>
<w id=mds0924>eyes</w>
<c id=mds0925>,</c>
<w id=mds0926>smile</w>
<c id=mds0927>,</c>
<w id=mds0928>and</w>
<w id=mds0929>wag</w>
<w id=mds0930>his</w>
<w id=mds0931>tail</w>
<w id=mds0932>as</w>
<w>
  <w id=mds0933>he</w>
  <m id=mds0934>'d</m>
  </w>
<w id=mds0935>done</w>
<w id=mds0936>on</w>
<w id=mds0937>one</w>
<w id=mds0938>occasion</w>
<w id=mds0939>when</w>
<w id=mds0940>confronted</w>
<w id=mds0941>by</w>
<w id=mds0942>a</w>
<w id=mds0943>housebreaker</w>
<c id=mds0944>!</c>
</s>
```

Finally, here is a <linkGrp>, which contains all of the <link> elements that associate the text segments in the example sentence with their respective grammatical structures.

```
<linkGrp domains='mds09 BNCgs'
         targFunc='segment analysis'
         extendTarg=0>
  <link targets='mds0901 AT0'>
  <link targets='mds0902 AJS'>
  <link targets='mds0903 PNP'>
  <link targets='mds0904 VVD'>
  <link targets='mds0905 PRP'>
  <link targets='mds0906 NN1'>
  <link targets='mds0907 VBD'>
  <link targets='mds0908 TO0'>
  <link targets='mds0909 VVI'>
  <link targets='mds0910 CRD'>
  <link targets='mds0911 NN1'>
  <link targets='mds0912 AV0'>
  <link targets='mds0913 PUN'>
  <link targets='mds0914 CJS'>
  <link targets='mds0915 PNI'>
  <link targets='mds0916 VVD'>
  <link targets='mds0917 AT0'>
  <link targets='mds0918 NN1'>
  <link targets='mds0919 PUN'>
  <link targets='mds0920 CJC'>
  <link targets='mds0921 TO0'>
  <link targets='mds0922 VVI'>
  <link targets='mds0923 DT0'>
  <link targets='mds0924 NN2'>
  <link targets='mds0925 PUN'>
  <link targets='mds0926 VVI'>
  <link targets='mds0927 PUN'>
  <link targets='mds0928 CJC'>
  <link targets='mds0929 VVI'>
  <link targets='mds0930 DPS'>
  <link targets='mds0931 NN1'>
  <link targets='mds0932 CJS'>
  <link targets='mds0933 PNP'>
  <link targets='mds0934 VHD'>
  <link targets='mds0935 VDN'>
  <link targets='mds0936 PRP'>
  <link targets='mds0937 CRD'>
  <link targets='mds0938 NN1'>
  <link targets='mds0939 AVQ'>
  <link targets='mds0940 VVN'>
  <link targets='mds0941 PRP'>
  <link targets='mds0942 AT0'>
  <link targets='mds0943 NN1'>
  <link targets='mds0944 PUN'>
  </linkGrp>
```

This grammatical markup represents the text as completely unambiguous, despite the fact that instances of the same textual unit are associated with different structure elements (e.g. the word 'to'), and at least one sequence (namely 'to exercise', identified by the attribute values id=mds0905 and id=mds0906), is in fact structurally ambiguous in English. That sequence may be analyzed as a preposition followed by a singular noun (as this markup asserts) or as the infinitive marker followed by an uninflected form of a main verb.

To represent the ambiguity of words like 'to' and 'exercise', and of phrases like 'to exercise', we may use the <alt> and <join> elements defined in sections 14.8 ('Alternation') on p. 373 and 14.7 ('Aggregation') on p. 369, as follows. First, we define <alt> elements for the ambiguous

word classes, and add these to the <fsLib>.

```
<alt id=PRP-TO0 targets='PRP TO0'>
<alt id=NN1-VVI targets='NN1 VVI'>
```

Next, we change the <link> elements for the text elements identified by the id=mds0905 and id=mds0905 attribute values.

```
<link targets='mds0905 PRP-TO0'>
<link targets='mds0906 NN1-VVI'>
```

As the encoding now stands, the phrase 'to exercise' has four structural analyses associated with it: preposition followed by noun, preposition followed by verb, infinitive marker followed by noun and infinitive marker followed by verb. To narrow the choices down to the desired two, namely preposition followed by noun and infinitive marker followed by verb, we next form <join> elements to represent the desired sequences.

```
  <join id=PRP.NN1 targets='PRP NN1'>
  <join id=TO0.VVI targets='TO0 VVI'>
```

We then define an <alt> element to express the alternation between the two <join> elements.

```
  <alt id=PRP.NN1-TO0.VVI targets='PRP.NN1 TO0.VVI'>
```

Next, we add a <phr> element in the encoding of the text for the phrase 'to exercise'.

```
<phr id=mds090506>
  <w id=mds0905>to</w>
  <w id=mds0906>exercise</w>
  </phr>
```

Finally, we add to the <linkGrp> element a <link> element connecting that phrase to the <alt> that represent its two analyses.

```
<link targets='mds090506 PRP.NN1-TO0.VVI'>
```

Note that the technique of forming <join> elements for sequences of structure elements and associating them with textual units can also be used to provide a complete structural analysis for the complex word 'he'd'. First, we add an id attribute for the word.

```
<w id=mds093334>
  <w id=mds0933>he</w>
  <m id=mds0934>'d</m>
  </w>
```

Next, we form a join of the structures associated separately with the subelements 'he' and ''d'.

```
<join id=PRP.VHD targets='PRP VHD'>
```

Finally, we define a link between the complex word and the new <join> element.

```
<link targets='mds093334 PRP.VHD'>
```

Chapter 17

Certainty and Responsibility

Encoders of text often find it useful to indicate that some aspects of the encoded text are problematic or uncertain, and to indicate who is responsible for various aspects of the markup of the electronic text. These Guidelines provide three methods of recording uncertainty about the text or its markup:

- the <note> element defined in section 6.8 ('Notes, Annotation, and Indexing') on p. 152 may be used with a value of `certainty` for its type attribute.
- the <certainty> element defined in this chapter may be used to record the nature and degree of the uncertainty in a more structured way.
- the <alt>element defined in the additional tag set for linking and segmentation may be used to provide alternative encodings for parts of a text, as described in section 14.8 ('Alternation') on p. 373.

There are three methods of indicating responsibility for different aspects of the electronic text:

- the TEI header records who is responsible for an electronic text by means of the <respStmt> element and other more specific elements (<author>, <sponsor>, <funder>, <principal>, etc.) used within the <titleStmt>, <editionStmt>, and <revisionDesc> elements.
- the <note> element may be used with a value of `resp` or `responsibility` in its type attribute.
- the <respons> element defined in this chapter may be used to record fine-grained structured information about responsibility for individual tags in the text.

To use the <note> and <respStmt> elements, no special steps are needed, since they are defined in the core tag set and header respectively. The <alt> element is only available when the additional tag set for linking has been selected, as described in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331. To use the <certainty> and <respons> elements, the additional tag set for certainty and responsibility must be selected; this is done by defining the parameter entity *tei.certainty* with the value `INCLUDE`, as shown in the example below:

```
<!DOCTYPE tei.2 system 'tei2.dtd' [
   <!ENTITY % tei.certainty 'INCLUDE' >
]>
 ...
```

17.1 Levels of Certainty

Many types of uncertainty may be distinguished. The <certainty> element is designed to encode the following sorts:

- a given tag may or may not correctly apply (e.g. a given word may be a personal name, or perhaps not)
- the precise point at which an element begins or ends is uncertain
- the value to be given for an attribute is uncertain

- content supplied by the encoder (such as the expansion of an abbreviation marked by the <abbr> tag) is uncertain
- the transcription of a source text is uncertain, perhaps because it is hard to read or hard to hear. (This sort of uncertainty is also handled by the <unclear> element in section 18.2.3 ('Damage, Illegibility, and Supplied Text') on p. 460)

The following types of uncertainty are *not* indicated with the <certainty> element:

- a number or date is imprecise
- the text is ambiguous, so a given passage has several possible interpretations
- a transcriber, editor, or author wishes to indicate a level of confidence in a factual assertion made in the text
- an author is not sure if the sentence she has chosen to start a paragraph is really the one she wants to retain in the final version

Precision of numbers and dates is discussed in section 6.4 ('Names, Numbers, Dates, Abbreviations, and Addresses') on p. 132; well-defined ambiguity is handled with alternations in feature-structure values in chapter 16 ('Feature Structures') on p. 397. Uncertainty about the truth of assertions in the text and other sorts of authorial and editorial uncertainty about whether the content is satisfactory are not handled by the <certainty> element, though they may be expressed using <note>.

17.1.1 Using Notes to Record Uncertainty

The simplest way of recording uncertainty about markup is to attach a note to the element or location about which one is unsure. In the following (invented) paragraph, for example, an encoder might be uncertain whether to mark "Essex" as a place name or a personal name, since both might be plausible in the given context:

"Elizabeth went to Essex. She had always liked Essex."

Using <note>, the uncertainty here may be recorded quite simply:

```
<persName>Elizabeth</> went to
<placeName>Essex</>.
She had always liked <placeName>Essex</>.
<note type='uncertainty' resp='MSM'>It is not
clear here whether <mentioned>Essex</mentioned>
refers to the place or to the nobleman. -MSM</note>
```

Using the normal mechanisms, the note may be associated unambiguously with specific elements of the text, thus:

```
<persName>Elizabeth</> went to
<placeName id=p1>Essex</>.
She had always liked <placeName id=p2>Essex</>.
<note type='uncertainty' resp='MSM' target='p1 p2'>
It is not clear here whether <mentioned>Essex</mentioned>
refers to the place or to the nobleman.  If the latter,
it should be tagged as a personal name. -MSM</note>
```

The advantage of this technique is its relative simplicity. Its disadvantage is that the nature and degree of uncertainty are not conveyed in any systematic way and thus are not susceptible to any sort of automatic processing.

17.1.2 Structured Indications of Uncertainty

To record uncertainty in a more structured way, susceptible of at least simple automatic processing, the <certainty> element may be used:

<certainty> indicates the degree of certainty or uncertainty associated with some aspect of the text markup. Attributes include:

target points at the elements whose markup is uncertain.

locus indicates the precise location of the uncertainty in the markup: applicability of the element, precise position of the start- or end-tag, value of a specific attribute, etc. Suggested values include:

#gi uncertain whether the element used actually applies to the passage.

#startloc start-tag may not be correctly located.

#endloc end-tag may not be correctly located.

#location both the start-tag and the end-tag may not be correctly located.

name the value given for the attribute name is uncertain.

#transcribedContent the content of the element may not be a correct transcription of the source text.

#suppliedContent the content of the element may not have been correctly supplied by the reader, e.g. as in the cases of <corr> and <abbrev> elements.

given indicates conditions assumed in the assignment of a degree of confidence.

degree indicates the degree of confidence assigned to the aspect of the markup named by the locus attribute.

assertedValue provides an alternative value for the aspect of the markup in question—an alternative generic identifier, transcription, or attribute value, or the ID of an <anchor> element (to indicate an alternative starting or ending location). If an assertedValue is given, the confidence level specified by degree applies to the alternative markup specified by assertedValue; if none is given, it applies to the markup in the text.

desc further describes the uncertainty in prose, perhaps indicating its nature, cause, or the justification for the degree of confidence asserted.

The <certainty> element may be used to record doubts about the proper encoding of "Essex" in several ways of varying precision. To record merely that we are not certain that "Essex" is in fact a place name, as it is tagged, we use the target attribute to identify the element in question, and the locus attribute to indicate what aspect of the markup we are uncertain about (here: whether we have used the correct element type):

```
Elizabeth went to <placeName id=p1>Essex</>.

<!-- ... elsewhere in the document ... -->
<certainty target=p1 locus='#gi' desc='possibly not a placename'>
```

Because it is linked to the location of the uncertainty by an IDREF, the <certainty> element will typically be included in the same SGML document as its target. It may be placed adjacent to the target element, or elsewhere in the document.

To record the further information that we estimate, subjectively, that there is a 60 percent chance of "Essex" being a place name here, we can add a value for our degree of confidence (usually a number between 0 and 1, representing the estimated probability):

```
Elizabeth went to <placeName id=p1>Essex</>.

<!-- ... -->
<certainty target=p1 locus='#gi'
           degree='0.6'
           desc='possibly not a placename'>
```

According to one expert, there is a 60 percent chance of "Essex" being a place name here, and a 40 percent chance of its being a personal name. We use two <certainty> elements to indicate the two probabilities independently. Both elements indicate the same location in the text, but the second provides an alternative choice of generic identifier (here: persName) is given as the value of the assertedValue attribute:

```
Elizabeth went to <placeName id=p1>Essex</>.

<!-- ... -->
<certainty target=p1 locus='#gi'
           degree='0.6'
           desc='probably a placename, but possibly not'>
<certainty target=p1 locus='#gi'
```

```
            degree='0.4'
            assertedValue='persName'
            desc='may refer to the Earl of Essex'>
```

Finally, we may wish to make our probability estimates contingent on some condition. In the passage "Elizabeth went to Essex; she had always liked Essex," for example, we may feel there is a 60 percent chance that the county is meant, and a 40 percent chance that the earl is meant. But the two occurrences of the word are not independent: there is (we may feel) no chance at all that one occurrence refers to the county and one to the earl. We can express this by using the given attribute to list the SGML identifiers of <certainty> elements.

```
Elizabeth went to <placeName id=p1>Essex</>.
She had always liked <placeName id=p2>Essex</>.

<!-- ... -->
<!-- 60% chance that P1 is a placename,
     40% chance a personal name. -->
<certainty id=cert-1
            target=p1 locus='#gi' degree='0.6'
            desc='probably a placename, but possibly not'>
<certainty id=cert-2
            target=p1 locus='#gi' degree='0.4'
            assertedValue='persName'
            desc='may refer to the Earl of Essex'>

<!-- 60% chance that P2 is a placename,
     40% chance a personal name.
     100% chance that it agrees with P1. -->
<certainty given=cert-1
            target=p2 locus='#gi' degree='1.0'
            desc='if P1 is a placename, P2 certainly is'>
<certainty given=cert-2
            target=p2 locus='#gi' degree='1.0'
            assertedValue='persName'
            desc='if p1 refers to the Earl of Essex, so does P2'>
```

When given conditions are listed, the <certainty> element is interpreted as claiming a given degree of confidence in a particular markup given the assertional content of the <certainty> elements indicated—that is, *if the markup described in the indicated <certainty> elements is correct.*

Conditional confidence may be less that 100 percent: given the sentence "Ernest went to old Saybrook", we may interpret "Saybrook" as a personal name or a place name, assigning a 60 percent probability to the former. If it is a place name, there may be a 50 percent chance that the place name actually in question is "Old Saybrook" rather than "Saybrook", while if it is correctly tagged as a personal name, it is much more likely (say, 90 percent certain) that the name is "Saybrook". This state of affairs can be expressed using the <certainty> element thus:

```
Earnest went to <anchor id=a1> old
<persName id=p1>Saybrook</persName>.

<certainty id=c1 target=p1 locus='#gi' degree='0.6'>
<certainty given=c1 target=p1 locus=startloc degree='0.9'>

<certainty id=c2 target=p1 locus='#gi'
            assertedValue='persName' degree='0.4'>
<certainty given=c2 target=p1 locus=startloc degree='0.5'>
<certainty id=c3
            given=c1 target=p1 locus=startloc
            assertedValue='a1' degree='0.5' >
```

In this case, the assertedValue on <certainty> element c3 is an IDREF to an <anchor> element at the alternate starting point for the element.

Multiplying the numeric values out, this markup may be interpreted as assigning specific probabilities to three different ways of marking up the sentence:

```
Earnest went to old <persName>Saybrook</>.   (0.6 * 0.9, or 0.54)
Earnest went to old <placeName>Saybrook</>.  (0.4 * 0.5, or 0.20)
Earnest went to <placeName>old Saybrook</>.  (0.4 * 0.5, or 0.20)
```

The probabilities do not add up to 1.00 because the markup indicates that if "Saybrook" is (part of) a personal name, there is a 10 percent likelihood that the element should start somewhere other than the place indicated, without however giving an alternative location; there is thus a 6 percent chance (0.1 * 0.6) that none of the alternatives given is correct.

If an attribute value is uncertain, the locus attribute takes as its value the name of the attribute in question. In this example, there is only a 50 percent chance that the question was spoken by participant A:

```
<u id=u1 who=A>Have you heard the election results?</u>
<!-- ... -->
<certainty target=u1 locus=who degree='0.5'>
```

Doubts about whether the transcription is correct may be expressed by assigning to locus the value #transcribedContent. For example, if the source is hard to read and so the transcription is uncertain:

```
I have a <emph id=p1>gub</emph>.
<certainty target=p1 locus='#transcribedContent' degree='0.5'>
```

Degrees of confidence in the proper expansion of abbreviations may also be expressed, by using the value #suppliedContent:

```
You will want to use <expan id=e1 abbr='SGML'>Standard
Generalized Markup Language</expan> ...

<!-- ... -->
<certainty target=e1 locus='#suppliedContent' degree='0.9'>
```

The assertedValue attribute should be used to provide an alternative value for whatever aspect of the markup is in doubt: an alternative generic identifier, or the ID of an alternative starting or ending point, as already shown, an alternative attribute value, or alternative element content, as in this example:

```
I have a <emph id=p1>gub</emph>.

<certainty target=p1 locus='#transcribedContent'
           assertedValue='gun'
           degree='0.8'
           desc='a gun makes more sense in a holdup'>
```

Since attribute values have no internal substructure, the assertedValue attribute is useful for specifying alternative transcriptions only in relatively restricted circumstances (specifically, when the alternate reading has no elements nested within it). More robust methods of handling uncertainties of transcription are the <unclear> element and the <app> and <rdg> elements described in chapter 19 ('Critical Apparatus') on p. 467. The <certainty> element allows for indications of uncertainty to be structured with at least as much detail and clarity as appears to be currently required in most ongoing text projects. It is expected that in the future more adequate systems for expressing uncertainty will be developed. These may extend the <certainty> element or they may make use of the feature-structure encoding mechanisms described in chapter 16 ('Feature Structures') on p. 397.

The <certainty> element and the other TEI mechanisms for indicating uncertainty provide a range of methods of graduated complexity. Simple expressions of uncertainty may be made by using the <note> element. This is simple and convenient, and can accommodate either discursive unstructured indication of uncertainty, or complex structured project-specific expressions of uncertainty. In general, however, unless special steps are taken, the <note> element does not provide as much expressive power as the <certainty> element, and in cases where highly structured certainty information must be given, it is recommended that the <certainty> element be used.

The <certainty> element may be used for simple unqualified indications of uncertainty, in which case only the locus and <target> might be specified. In more complex cases, the other

attributes may be used to provide fuller information. While they may take any string of characters as value, the recommended values should be used wherever possible; if they are not appropriate in a given situation, encoders are should provide their own controlled vocabulary and document it in the <encodingDesc> or <tagUsage> elements of the TEI header.

The <certainty> element has the following formal declaration:

```
<!-- 17.1.2:  Certainty and uncertainty                          -->
<!-- Text Encoding Initiative: Guidelines for Electronic         -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.       -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy       -->
<!-- in any form is granted, provided this notice is             -->
<!-- included in all copies.                                     -->

<!-- These materials may not be altered; modifications to        -->
<!-- these DTDs should be performed as specified in the          -->
<!-- Guidelines in chapter "Modifying the TEI DTD."              -->

<!-- These materials subject to revision. Current versions       -->
<!-- are available from the Text Encoding Initiative.            -->

<!ELEMENT certainty     - O  EMPTY                                >
<!ATTLIST certainty          %a.global;
          target             IDREFS              #REQUIRED
          locus              CDATA               #REQUIRED
          assertedValue      CDATA               #IMPLIED
          desc               CDATA               #IMPLIED
          given              CDATA               #IMPLIED
          degree             CDATA               #IMPLIED      >
<!-- ... declarations from section 17.2                          -->
<!--     (Responsibility for markup)                             -->
<!--     go here ...                                             -->
```

17.2 Attribution of Responsibility

In general, attribution of responsibility for the transcription and markup of an electronic text is made by <respStmt> elements within the header: specifically, within the title statement, the edition statement(s), and the revision history.

In some cases, however, more detailed element-by-element information may be desired, in order to distinguish, for example, between the individuals responsible for transcribing the content and those responsible for determining that a given word or phrase constitutes a proper noun. Where such fine-grained attribution of responsibility is required, the <respons> element may be used:

<respons> identifies the individual(s) responsible for some aspect of the markup of some particular element(s). Attributes include:

- **target** gives the SGML identifier(s) of the element(s) for which some aspect of the responsibility is being assigned.
- **locus** indicates the specific aspect of the markup for which responsibility is being assigned. Suggested values include:
  - ***#gi*** responsibility for the claim that the element is of the type indicated by the markup
  - ***#location*** responsibility for the claim that the element begins and ends where indicated
  - ***#startloc*** responsibility for the claim that the element begins where indicated
  - ***#endloc*** responsibility for the claim that the element ends where indicated
  - ***name*** responsibility for the claim that the name attribute has the value given in the markup
  - ***#transcribedContent*** responsibility for the transcription of the element content

#suppliedContent responsibility for the contents supplied by the encoder (corrections, expansions of abbreviations, etc.)

resp identifies the individual or agency responsible for the indicated aspect of the electronic text.

desc gives a brief prose note supplying any additional information which should be recorded

This element allows one or more aspects of the markup to be attributed to a given individual. The target and locus attributes function as they do on the <certainty> element described in section 17.1 ('Levels of Certainty') on p. 435: the target attribute points at a particular SGML element (or set of elements), and locus indicates the particular aspect of the encoding of those elements, for which responsibility is to be assigned. The suggested values may be combined as appropriate: to indicate, for example, that RC is responsible for transcribing an illegible word, and that AR is responsible for identifying that word as a proper noun, the text might be encoded thus:

```
Earnest went to old <persName id=p1>Saybrook</persName>.
<!-- ... -->
<respons target=p1 locus='#transcribedContent' resp=RC>
<respons target=p1 locus='#gi #location' resp=AR>
```

Some elements bear specialized resp or agent attributes, which have specific meaning which varies from element to element; the <respons> element should be reserved for the general aspects of responsibility common to all text transcription and SGML markup, and should not be confused with the more specific attributes on individual elements.

The formal declaration of the <respons> element is this:

```
<!-- 17.2:  Responsibility for markup                          -->
<!ELEMENT respons       - O  EMPTY                              >
<!ATTLIST respons            %a.global;
          target             IDREFS              #REQUIRED
          locus              CDATA               #REQUIRED
          resp               CDATA               #REQUIRED
          desc               CDATA               #IMPLIED      >
<!-- This fragment is used in sec. 17.1.2                      -->
```

Chapter 18

Transcription of Primary Sources

This chapter defines an optional additional tag set intended for use in the transcription of primary sources, in particular manuscripts, and describes how some elements defined in the core tag set should be used for this work. It is expected that this tag set will be especially useful in the preparation of critical editions, but the tag set defined here is distinct from that defined in chapter 19 (‘Critical Apparatus’) on p. 467, and may be used independently of it.

Scholars may wish to record information concerning individual readings of letters, words or larger units, both within transcriptions and within editions. They may also wish to include other editorial material within transcriptions, such as comments on the status or possible origin of particular readings, corrections, or text supplied to fill lacunae, etc. Further, it is customary in transcriptions to register certain features of the source, such as ornamentation, underlining, deletion, areas of damage and lacunae. This chapter indicates means to record such information:

- first, the problem of recording editorial or other alterations to the text, such as expansion of abbreviations, corrections, conjectures, etc. (section 18.1 (‘Altered, Corrected, and Erroneous Texts’) on p. 445)
- then, methods of describing important extra-linguistic phenomena in the source: unusual spaces, lines, page and line breaks, change of manuscript hand, etc. (section 18.2 (‘Non-Linguistic Phenomena in the Source’) on p. 456)
- finally, a method of recording material such as running heads, catch-words, and the like (section 18.3 (‘Headers, Footers, and Similar Matter’) on p. 465)

These recommendations are not intended to meet every transcriptional circumstance likely to be faced by any scholar. Rather, they should be regarded as a base which can be elaborated if necessary by different scholars in different disciplines, with distinct scholarly domains eventually developing their own document types. In time, the feature structure notation developed in chapter 16 (‘Feature Structures’) on p. 397, may also permit scholars to tailor the encoding of complex transcriptional information in ways not here anticipated. In particular, this chapter focuses in its current state primarily upon problems associated with the transcription of manuscript materials; problems of codicology and problems peculiar to early printed materials are not treated. Many of the recommendations presented here may — mutatis mutandis — apply to printed matter, but a great deal of work remains to done in these areas, and the encoder will need to take even more individual responsibility than usual in applying the recommendations of this chapter in these contexts.

Many of the descriptions below use terms like “scribe”, “author”, “editor”, “annotator”, “corrector”, “transcriber”, and “encoder”, to make clear how they apply in cases where these roles are distinct. To the extent that these roles are not distinct (for example, in authorial manuscripts where the author and the scribe are the same person) the interpretation of the markup should be adjusted appropriately. Many of the elements defined here apply (within limits) also in cases of printed materials, so “compositor”, etc., may also be understood as applying where appropriate.

As a rule, all elements which may be used in the course of a transcription of a single witness may also be used in a critical apparatus, i.e. within the elements proposed in chapter 19 ('Critical Apparatus') on p. 467. This can generally be achieved by nesting a particular reading containing tagged elements from a particular witness within the <rdg> element in an <app> structure.

Just as a critical apparatus may contain transcriptional elements within its record of variant readings in various witnesses, one may record variant readings in an individual witness by use of the apparatus mechanisms <app> and <rdg>. This is discussed in section 19.3 ('Using Apparatus Elements in Transcriptions') on p. 485.

The tag set defined in this chapter may be selected using the mechanisms described in section 3.3 ('Invocation of the TEI DTD') on p. 39; in a document using this tag set, the document-type-declaration subset should contain the following declaration of the parameter entity *TEI.transcr*, or the equivalent:

```
<!ENTITY % TEI.transcr 'INCLUDE' >
```

In a document using this tag set together with that for textual criticism and the base tag set for verse, the entire document type declaration might resemble the following:

```
<!DOCTYPE tei.2 system 'tei2.dtd' [
   <!ENTITY % TEI.prose 'INCLUDE' >
   <!ENTITY % TEI.transcr 'INCLUDE' >
   <!ENTITY % TEI.textcrit 'INCLUDE' >
]>
```

The overall structure of the tag set defined by this chapter is as follows:

```
<!-- 18:  Transcription of Primary Sources                  -->
<!-- Text Encoding Initiative: Guidelines for Electronic    -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.  -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy  -->
<!-- in any form is granted, provided this notice is        -->
<!-- included in all copies.                                -->

<!-- These materials may not be altered; modifications to   -->
<!-- these DTDs should be performed as specified in the     -->
<!-- Guidelines in chapter "Modifying the TEI DTD."         -->

<!-- These materials subject to revision. Current versions  -->
<!-- are available from the Text Encoding Initiative.       -->
<!-- ... declarations from section 18.1.4                   -->
<!--     (Added and Deleted Spans)                          -->
<!--     go here ...                                        -->
<!-- ... declarations from section 18.1.6                   -->
<!--     (Cancelled Deletions)                              -->
<!--     go here ...                                        -->
<!-- ... declarations from section 18.1.7                   -->
<!--     (Supplied Text)                                    -->
<!--     go here ...                                        -->
<!-- ... declarations from section 18.2.1                   -->
<!--     (Hand Shifts)                                      -->
<!--     go here ...                                        -->
<!-- ... declarations from section 18.2.3                   -->
<!--     (Damage and Illegiblity)                           -->
<!--     go here ...                                        -->
<!-- ... declarations from section 18.2.5                   -->
<!--     (Spaces in the source)                             -->
<!--     go here ...                                        -->
<!-- ... declarations from section 18.3                     -->
<!--     (Headers and footers)                              -->
<!--     go here ...                                        -->
```

This tag set modifies the element class *edit* by declaring two extra attributes for members of the class:

```
<!-- 18:  Attributes for Transcription of Primary Sources     -->
<!-- Text Encoding Initiative: Guidelines for Electronic      -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.    -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy    -->
<!-- in any form is granted, provided this notice is          -->
<!-- included in all copies.                                  -->

<!-- These materials may not be altered; modifications to     -->
<!-- these DTDs should be performed as specified in the       -->
<!-- Guidelines in chapter "Modifying the TEI DTD."           -->

<!-- These materials subject to revision. Current versions    -->
<!-- are available from the Text Encoding Initiative.         -->
<!ENTITY % a.edit '
          resp               IDREF              %INHERITED
          cert               CDATA              #IMPLIED'      >
```

18.1 Altered, Corrected, and Erroneous Texts

In the detailed transcription of any source, it may prove necessary to record various types of actual or potential alteration of the text: expansion of abbreviations, correction of the text (by the author, by a scribe, by a later hand, or by the encoder), addition, deletion, or substitution of material, and the like. The sections below describe how such phenomena may be encoded using either elements defined in the core tag set (defined in chapter 6 ('Elements Available in All TEI Documents') on p. 119) or specialized elements available only when the additional tag set described in this chapter is available.

18.1.1 Use of Core Tags for Transcriptional Work

In transcribing individual sources (editions, manuscripts, witnesses of any type), encoders may record their corrections, normalizations, expansions of abbreviations, additions, and omissions using the elements described in section 6.5 ('Simple Editorial Changes') on p. 140. Those particularly relevant to this chapter include:

<abbr> contains an abbreviation of any sort.
<expan> contains the expansion of an abbreviation.
<sic> contains text reproduced although apparently incorrect or inaccurate.
<corr> contains the correct form of a passage apparently erroneous in the copy text.
<add> contains letters, words, or phrases inserted in the text by an author, scribe, annotator or corrector.
<del> contains a letter, word or passage deleted, marked as deleted, or otherwise indicated as superfluous or spurious in the copy text by an author, scribe, annotator or corrector.
<hi> marks a word or phrase as graphically distinct from the surrounding text, for reasons concerning which no claim is made.
<gap> indicates a point where material has been omitted in a transcription, whether for editorial reasons described in the TEI header, as part of sampling practice, or because the material is illegible or inaudible.

When the additional tag set for transcription of primary sources is selected, these elements all gain two specialized attributes for specifying who is responsible for certain aspects of the interpretation and markup, and the certainty attributed to the interpretation:

cert signifies the degree of certainty ascribed to some specific aspect of the markup: the identification of the hand of an addition or deletion, the correctness of the expansion of an abbreviation, the correction of an error, or the regularization of a non-standard form; or the correctness of the transcription of unclear material.

resp signifies the editor or transcriber responsible for the salient information conveyed by a particular tag: the hand of an addition or deletion, the expansion of an abbreviation, the correction of an apparent error, the regularization of a non-standard form, the transcription of unclear material, or the decision not to transcribe some portion of the text.

The specific aspect of the markup described by these attributes differs on different elements; for further discussion, see the relevant sections below, especially section 18.2.2 ('Hand, Responsibility, and Certainty Attributes') on p. 459.

The following sections describe how the core elements just named may be used in the transcription of primary source materials. Examples of more complex application in scholarly transcriptions of these core elements are given, and of their extension by linkage with the <note>, <respons>, and <certainty> elements. Where the core elements do not satisfy the needs of scholarly transcription, additional elements are defined.

18.1.2 Abbreviation and Expansion

The writing of manuscripts by hand lends itself to the use of abbreviation to shorten scribal labour. Commonly occurring letters, groups of letters, words or even whole phrases, may be represented by significant marks. This phenomenon of manuscript abbreviation is so widespread and so various that no taxonomy of it is here attempted. Instead, methods are shown which allow abbreviations to be encoded using the core elements mentioned above.

A manuscript abbreviation may be viewed in two ways. One may transcribe it as a particular sequence of letters or marks upon the page: thus, a "p with a bar through the descender", a "superscript hook", a "macron". One may also interpret the abbreviation in terms of the letter or letters it is seen as standing for: thus, "per", "re", "n". Both of these views are supported by these Guidelines. The entity reference system allows the encoder to declare whatever entities are needed, using entity names like *p-underbar*, *sup-hook*, or *macron*. Furthermore, each entity reference may be linked to an image of the abbreviation itself, so that the reader might see a rendering of the text's appearance. Alternatively, the encoder may transcribe the letter or letters he or she believes the abbreviation stands for, as the content of an <expan> element: thus <expan>per</>, <expan>re</>, <expan>n</>.

These two methods of coding abbreviation may also be combined. An encoder may record, for any abbreviation, both the sequence of letters or marks which constitutes it, and its sense, that is, the letter or letters for which it is believed to stand. For example, the abbreviations of 'euery persone' in the following fragment [1] may be transcribed as follows, using the <expan> element, with the abbr attribute to hold an entity reference for the brevigraph indicating the abbreviation in the manuscript:

```
eu<expan abbr="&er;" resp="MP">er</>y
<expan abbr="&p-underbar">per</>sone that
loketh after heuen hath a place in this ladder
```

Alternatively, the abbreviations may be encoded using the <abbr> element.

```
eu<abbr expan="er" resp=MP>&er;</>y
<abbr expan="per">&p-underbar</>sone that
loketh after heuen hath a place in this ladder
```

The choice between the <expan> and <abbr> elements is left to the encoder. As a rule, the <abbr> element should be preferred where it is wished to signify that the content of the element is an abbreviation, without necessarily indicating what the abbreviation may stand for. The <expan> element should be used where it is wished to signify that the content of the element is an expanded text, without necessarily indicating the abbreviation used in the original. The decision as to which (<abbr> or <expan>) to use may vary from abbreviation to abbreviation; there is no requirement that the one system be used throughout a transcription. However, processing may be simplified if one only of these is used throughout a transcription. The choice is likely to be a matter of editorial policy, which might be applied consistently throughout. If the highest priority is to transcribe the text literatim, while indicating the presence of abbreviations, the choice will

[1] From "The Manere of Good Lyuynge", fol 126v of Bodleian MS Laud Misc. 517, plate 8(ii) in *English Cursive Book Hands 1250-1500* by M. B. Parkes (Clarendon Press: Oxford, 1969).

be to use <abbr> throughout. If the highest priority is to present a reading transcription, while indicating that some letters or words are expansions of abbreviations, the choice will be to use <expan> throughout.

Further information may be attached to instances of these elements by the <note> element, on which see section 6.8 ('Notes, Annotation, and Indexing') on p. 152, and by use of the resp and cert attributes. In this instance from the English *Brut*,[2] a note is attached to an editorial expansion of the tail on the final d of 'good' to 'goode':

```
For alle the while that I had
good<expan id=exp01 abbr="&tail;">e</>
I was welbeloued
```

Then the note:

```
<note target=exp01>The stroke added to the final d could
signify the plural ending (-es, -is, -ys>) but the
singular <hi rend=it>good</> was used with the meaning
<q>property</q>, <q>wealth</q>, at this time (v. examples
quoted in OED, sb. Good, C. 7, b, c, d and 8 spec.)</note>
```

The editor might declare a degree of certainty for this expansion, based on the OED examples, and state the responsibility for the expansion:

```
For alle the while that I had
good<expan abbr="&tail;"
           cert=90
           resp="MP">e</>
I was welbeloued
```

Observe that the cert and resp attributes may be used with the <expan> element only to indicate respectively confidence in the content of the element, (i.e., the expansion) and the responsibility for suggesting this expansion. In the case of the use of these attributes with the <abbr>, the cert and resp attributes are defined as indicating respectively confidence in the expansion held in the expan attribute and the responsibility for suggesting this expansion. The above example could be encoded using the <abbr> element as follows:

```
For alle the while that I had
good<abbr expan="e"
          cert=90
          resp="MP">&tail;</>
I was welbeloued
```

If it is desired to express aspects of certainty and responsibility for some other aspect of the use of these elements, then the mechanisms discussed in chapter 17 ('Certainty and Responsibility') on p. 435 should be used. See also 18.2.2 ('Hand, Responsibility, and Certainty Attributes') on p. 459 for discussion of the issues of certainty and responsibility in the context of transcription.

If more than one expansion for the same abbreviation is to be recorded, it is recommended that the markup for critical apparatus be used; an example is given in section 19.3 ('Using Apparatus Elements in Transcriptions') on p. 485.

18.1.3 Correction and Conjecture

The <sic> and <corr> elements, defined in the core tag set, may be used to register authorial or scribal corrections within a witness. For example, in the manuscript of William James's *A Pluralistic Universe*, edited by Fredson Bowers (Cambridge: Harvard University Press, 1977) a sentence first written

"One must have lived longer with this system, to appreciate its advantages."

has been modified by James to begin "But one must ...", without the inital capital O having been reduced to lowercase. This non-standard orthography could be recorded and corrected thus:

```
But <sic corr="one">One</> must have lived ...
```

[2] On fol 65v of Bodleian MS. Rawlinson Poetry 32; in Parkes 12(ii).

The same information could be conveyed by the <corr> element:

```
But <corr sic="One">one</> must have lived ...
```

In this example from Albertus Magnus,[3] both the manuscript error 'angues' and its correction 'augens' are registered by the <sic> element:

```
Nos autem iam ostendimus quod nutrimentum
et <sic corr=augens>angues</>.
```

The same information could be conveyed by the <corr> element:

```
Nos autem iam ostendimus quod nutrimentum
et <corr sic="angues">augens</>.
```

In this example, from George Moore's draft of additional materials for "Memoirs of My Dead Life"[4] the transcriber supplies the word 'we' omitted by the author:

```
You see that I avoid the word create for we
create nothing <sic corr="we"></> develope.
```

Or with reverse use of the <corr> element:

```
You see that I avoid the word create for we
create nothing <corr sic="">we</> develope.
```

(N.B. when the additional tag set defined in this chapter is selected, the <supplied> element should normally be used in preference to <sic> or <corr> for such supplied text.)

As with the choice between <expan> and <abbr>, the choice between the synonymous <sic> and <corr> elements is left to the encoder. As a rule, the <sic> element allows the encoding to retain the original text as the content of the element, while simultaneously signifying that the contents of the element require correction, but without necessarily indicating what the correction may be. The <corr> element allows the text to be corrected, possibly without recording the details of the faulty source, while still marking explicitly the fact that the contents of the element have been corrected. The choice is likely to be a matter of editorial policy, which might be applied consistently throughout or decided case by case. If the highest priority is to present an uncorrected transcription while noting perceived errors in the original, the choice will typically be to use <sic> throughout. If the highest priority is to present a reading transcription, while indicating that perceived errors in the original have been corrected, the choice will be to use <corr> throughout.

Further information may be attached to instances of these elements by the <note> element and resp and cert attributes. Here, two separate corrections in Dudo of S. Quentin,[5] are assigned the same note. First the corrections, held in the attribute value of the <sic> elements:

```
quamuis <sic corr=iners id=sic01>mens</> que nutu dei
gesta sunt ... unde esset uiriliter
<sic corr=uegetata id=sic02>negata</>
```

then the note, linked to the id of the <sic> element for each of the two corrections:

```
<note target="sic01 sic02">Substitution of a more
familiar word which resembles graphically what the
scribe should be copying but which
does not make sense in the context.</>
```

The cert attribute may also be used with the <corr> element to signify the conjectural status of a particular editorial reading, with the resp attribute used to identify the scholar responsible for the conjecture. In this example, editorial confidence in E. Talbot Donaldson's emendation of the Hengwrt manuscript reading 'wight' to 'wright' in line 117 of Chaucer's *The Wife of Bath's Prologue* may be marked as follows:

```
Telle me also, to what conclusioun
Were membres maad, of generacioun
And of so parfit wis a <corr sic="wight" resp=ETD cert=70
id="c117">wright</> ywroght?
```

[3] *De Nutrimento et Nutribili, Tractatus 1*, fol 217r col b of Merton College Oxford MS O.2.1, (Parkes pl. 16).

[4] In Pierpont Morgan MA 3421, from *British Literary Manuscripts/Series II: from 1800 to 1914*, by V. Klinkenborg, H.Cahoon and C. Ryskamp (New York: Pierpont Morgan Library, 1981).

[5] *De moribus et actis primorum Normannie ducum*, in fol 4v of British Library MS Harley 3742, Parkes pl 6(i).

The editor might also conveniently add a note referring to Donaldson's discussion of this passage:

```
<note target=c117>This emendation of the Hengwrt copy text,
based on a Latin source and on the reading of three late
and usually unauthoritative manuscripts, was proposed
by E. Talbot Donaldson in <title>Speculum</title> 40 (1965)
626-33.</note>
```

Alternative corrections within a transcription of a single witness may be held within an <app> structure, in the same way that alternative expansions are so grouped in the example given in section 19.3 ('Using Apparatus Elements in Transcriptions') on p. 485. Here, Donaldson's conjectured emendation of the Hengwrt manuscript may be recorded not only alongside the editorial transcription but also alongside another conjecture:

```
And of so parfit wis a
<app>
   <rdg wit=Hg>wight</rdg>
   <rdg resp=ETD wit="Ln Ry2 Ld"><corr>wright</></rdg>
   <rdg resp=PR wit="Gg"><corr>wyf</></rdg>
</app>
```

Observe that no resp attribute is necessary for the base transcription: by default, responsibility is assigned to the scholar(s) responsible for the transcription, as identified in the TEI header. The conjectures are held within <corr> elements, contained within the <rdg> elements. The resp attribute identifying responsibility for each correction is attached to the outer <rdg>, and inherited by the inner <corr> element. Note too that the support for these conjectures in other manuscripts can be noted in the wit attribute in the <rdg> element.

The cert and resp attributes may be used with the <corr> element only to indicate respectively confidence in the content of the element, (i.e., the correction) and the responsibility for suggesting this correction or conjecture. In the case of the use of these attributes with the <sic> element, the cert and resp attributes are defined as indicating respectively confidence in the conjecture held in the corr attribute and the responsibility for suggesting this conjecture. The above example could be encoded using the <sic> element as follows:

```
And of so parfit wis a
<sic corr="wright" cert=70 resp=ETD>wight</>
ywroght?
```

If it is desired to express aspects of certainty and responsibility for some other aspect of the use of these elements, then the mechanisms discussed in chapter 17 ('Certainty and Responsibility') on p. 435 should be used. See also 18.2.2 ('Hand, Responsibility, and Certainty Attributes') on p. 459 for discussion of the issues of certainty and responsibility in the context of transcription.

18.1.4 Additions and Deletions

Additions and deletions to a text may be described using the following elements:

<add> contains letters, words, or phrases inserted in the text by an author, scribe, annotator or corrector.

<addSpan> marks the beginning of a longer sequence of text added by an author, scribe, annotator or corrector (see also <add>). Attributes include:

place indicates where the addition is made. Suggested values include:

- ***inline*** addition is made in a space left in the witness by an earlier scribe.
- ***supralinear*** addition is made above the line.
- ***infralinear*** addition is made below the line.
- ***marginleft*** addition is made in left margin.
- ***marginright*** addition is made in right margin.
- ***margintop*** addition is made in top margin.
- ***marginbot*** addition is made in bottom margin.

overleaf addition is made on the other side of the leaf.

resp signifies the editor or transcriber responsible for identifying the hand of the addition.

cert signifies the degree of certainty ascribed to the identification of the hand of the addition.

hand signifies the hand of the agent which made the addition.

to identifies the endpoint of the added passage, by giving the ID of an <anchor> or other empty element placed there.

<del> contains a letter, word or passage deleted, marked as deleted, or otherwise indicated as superfluous or spurious in the copy text by an author, scribe, annotator or corrector.

<delSpan> marks the beginning of a longer sequence of text deleted, marked as deleted, or otherwise signaled as superfluous or spurious by an author, scribe, annotator, or corrector. Attributes include:

type classifies the deletion, using any convenient typology. Sample values include:

overstrike deletion indicated by line crossing out the text.

erasure deletion indicated by erasure of the text.

bracketed deletion indicated by brackets in the text or margin.

subpunction deletion indicated by dots beneath the letters deleted.

status indicates whether the deletion is faulty, e.g. by including too much or too little text. Sample values include:

excess start some text at the beginning of the deletion is marked as deleted even though it clearly should not be deleted.

excess end some text at the end of the deletion is marked as deleted even though it clearly should not be deleted.

short start some text at the beginning of the deletion is not marked as deleted even though it clearly should be.

short end some text at the end of the deletion is not marked as deleted even though it clearly should be.

unremarkable the deletion is not faulty.

resp signifies the editor or transcriber responsible for identifying the hand of the deletion.

cert signifies the degree of certainty ascribed to the identification of the hand of the deletion.

hand signifies the hand of the agent which made the deletion.

to identifies the endpoint of the deleted passage, by giving the ID of an <anchor> or other element placed there.

Of these, <add> and <del> are included in the core tag set, while <addSpan> and <delSpan> are available only when using the additional tag set defined in this chapter.

As described in section 6.5 ('Simple Editorial Changes') on p. 140, the <add> element indicating material added may be used to signify manuscript additions or insertions, be they authorial or scribal. In the autograph manuscript of Max Beerbohm's *The Golden Drugget*,[6] the author's addition of "do ever" may be recorded as follows, with the hand attribute indicating that the addition was Beerbohm's:

```
Some things are best at first sight.  Others -- and
here is one of them -- <add hand='MB'>do ever</add>
improve by recognition
```

Similarly, the <del> element indicating material deleted may be used to signify manuscript deletions. In the autograph manuscript of D. H. Lawrence's *Eloi, Eloi, lama sabachthani* (Pierpont Morgan MA 1892, Klinkenborg 129), the author's deletion of 'my' may be recorded as follows. As well as the hand attribute indicating that the deletion was Lawrence's, the rend attribute indicates that the deletion was by strike-through:

```
For I hate this
<del hand='DHL' rend='strike-through'>my</del> body,
which is so dear to me
```

If deletions are classified systematically, the type attribute should normally be used to indicate the classification; when they are classified by the manner in which they were effected, or by their

[6] In Pierpont Morgan MA 3391 (Klinkenborg 123).

appearance, however, this will lead to a certain arbitrariness in deciding whether to use the type or the rend attribute to hold the information. In general, it is recommended that the rend attribute be used for description of the appearance or method of deletion, and that the type attribute be reserved for higher level or more abstract classifications.

Further characteristics of the addition and deletion, e.g. the date, or ink, may be needed for detailed transcription of manuscripts. Such characteristics may conveniently be recorded as attributes of the <add> or <del> element. The specific attributes required may be added to the formal declaration of these elements by using the techniques described in chapter 29 ('Modifying the TEI DTD') on p. 619.

The <add> and <del> elements defined in the core tag set available in all TEI documents will suffice for describing typically brief additions and deletions in the text being transcribed. On occasion, it will be necessary to record an addition or deletion which crosses a structural boundary in the text being encoded, for example the addition or deletion from a manuscript of a section containing several distinct structural subdivisions, such as poems or prose items. These are most conveniently encoded using the <addSpan> and <delSpan> elements, available in the additional tag set defined in this chapter. In this example of the use of <addSpan>, the insertion of a gathering containing four neo-Eddic poems into Landsbókasafn (Reykjavík, 1562 quarto) by Helgi Ólafsson is recorded as follows. The <addSpan> element is placed at the beginning of the span of added text. The hand attribute ascribes the responsibility for the addition to the manuscript to Helgi, and the to attribute declares the identifier for the anchor which marks the end of the added text:

```
<!-- text of the original material ... -->

<addSpan type="added gathering"
         hand="Helgi &Oacute;lafsson"
         to=p025>
<!-- text of the four neo-Eddic poems added... -->
<anchor id=p025>

<!-- text of the original material continues... -->
```

In this example of the use of the <delSpan> element, a full two lines of Thomas Moore's autograph of the second version of *Lalla Rookh*[7] are marked for omission by vertical strike-through. The two lines cross the structural line division marked <l n=2>, so it would not be possible to use a single <del> element, since it would have to span the <l> marker. The lines also themselves include a further deletion and addition. The <delSpan> element indicates the begining of the span marked for deletion, with the to attribute giving the identifier (delend01) for the anchor which marks the end of the span of text so marked:

```
<l n=1>
<delSpan type='vertical strike' to=delend01>
Tis moonlight <del>upon</><add>over</> Oman's sky
<l n=2> Her isles of pearl look lovelily
<anchor id=delend01>
</l>
```

The text deleted must be at least partially legible, in order for the encoder to be able to transcribe it. If it is not legible at all, the <gap> element should be used to signal that the text was not (because it could not be) transcribed; the reason attribute can give the cause of the omission from the transcription as "deletion, illegible". The <gap> element may optionally be enclosed by a <del> element, if it is thought useful to record the deletion explicitly using this element. If the deleted text is partially legible, the <unclear> element described in section 18.2.3 ('Damage, Illegibility, and Supplied Text') on p. 460 should be used to signal the areas of text which cannot be read with confidence; it too may be enclosed within a <del> element. See further section 18.1.7 ('Text Omitted from or Supplied in the Transcription') on p. 455 and section 18.2.3 ('Damage, Illegibility, and Supplied Text') on p. 460.

The elements <add>, <del>, and <gap> are defined in the core tag set and are available in all TEI documents. The elements <addSpan> and <delSpan> have the following formal

[7] In Pierpont Morgan MA 310, (Klinkenborg 23).

declarations:

```
<!-- 18.1.4:  Added and Deleted Spans                          -->
<!ELEMENT addSpan       - O  EMPTY                              >
<!ATTLIST addSpan            %a.global;
          type               CDATA              #IMPLIED
          place              CDATA              #IMPLIED
          resp               IDREF              %INHERITED
          cert               CDATA              #IMPLIED
          hand               IDREF              %INHERITED
          to                 IDREF              #REQUIRED      >
<!ELEMENT delSpan       - O  EMPTY                              >
<!ATTLIST delSpan            %a.global;
          type               CDATA              #IMPLIED
          resp               IDREF              %INHERITED
          cert               CDATA              #IMPLIED
          hand               IDREF              %INHERITED
          to                 IDREF              #REQUIRED
          status             CDATA              'unremarkable' >
<!-- This fragment is used in sec. 18                          -->
```

18.1.5 Substitutions

Substitution of one word or phrase for another is perhaps the most common of all phenomena requiring special treatment in transcription of primary textual sources. It may be simply one word overwriting another, or deletion of one word and its replacement by another written above it by the same hand at the one time; the deletion and replacement may be done by different hands at different times; there may be a long chain of substitutions on the one stretch of text, with uncertainty as to the order of substitution and as to the final reading.

Three different methods may be used to express substitution of one stretch of text by another:

- the <sic> and <corr> elements, either individually to encode a single substitution or nested to encode a sequence of substitutions;
- the <del> and <add> elements, used in sequence to show that text was first deleted then other text inserted;
- the <del> and <add> elements, used within an <app> structure (as defined in chapter 19 ('Critical Apparatus') on p. 467) to indicate that the deleted and added text within the individual reading elements making up the <app> structure are variants of one another.

The use of all three of these is illustrated in the following encodings of the second line of *Eloi, Eloi, lama sabachthani* from the Lawrence manuscript mentioned above. Lawrence first wrote "How it galls me, what a galling shadow". Subsequently, he deleted 'galls' and wrote 'dogs' above the deletion.

This substitution could be registered using the first method outlined above, as a correction using the <sic> or <corr> elements. Note the use of the resp attribute on the <corr> element to assign the correction to Lawrence. (For further information on the hand and resp attributes, see section 18.2.2 ('Hand, Responsibility, and Certainty Attributes') on p. 459.)

```
How it <corr resp='DHL' sic='galls'>dogs</corr>
me, what a galling shadow
```

This substitution could be registered using the second method outlined above, using the <del> and <add> elements in sequence to reflect the fact that text was first deleted then other text inserted:

```
How it <del type=overstrike hand='DHL'>galls</del>
<add place=supralinear hand='DHL'>dogs</add>
me, what a galling shadow
```

This substitution could be registered using the third method outlined above, using the <del> and <add> elements within an <app> structure to indicate that the deleted and added texts are variants of one another. Note that within the <app> structure the hand attribute is moved from the inner <del> and <add> elements to the outer <rdg> element:

```
How it
  <app>
     <rdg hand='DHL'><del type=overstrike> galls</del></rdg>
     <rdg hand='DHL'><add place=superlinear> dogs</add> </rdg>
  </app>
me, what a galling shadow
```

Each of these three methods has its particular advantages and disadvantages. The first method (use of <sic> or <corr>) is compact and indicates clearly that one text is a substitute for another. However, it provides no clear means of stating how the substitution is effected: whether by deletion through strike-out, or underdotting, or erasure, followed by interlinear insertion, or marginal insertion. (The global rend attribute might conceivably be used, but this may not be thought an obvious place to put such information.) In a transcription where this information is not felt to be important, however, this method will suffice to indicate simple cases of direct substitution of one text for another.

The second method (use of a <del> and <add> sequence) is also compact and provides means for exact declaration of how the deletion and insertion are effected. However, it does not indicate explicitly that one text is a substitute for another. It is left for the reader or the application to infer from the <del> and <add> sequence that the insertion is to be taken as a substitution for the deletion. In many transcriptions, the inference may be safely drawn for simple cases of direct substitution of one text for another. In other transcriptions, for example of complex authorial manuscripts, this inference may prove fragile; those who desire to express clearly that an adjacent addition and deletion are not independent but constitute a single act of substitution will therefore wish to avoid this method. Others, of course, may prefer it for precisely the same reason, namely that it avoids prejudging the issue of whether adjacent deletions and additions are independent or joined.

The third method (use of the <del> and <add> elements within an <app> structure) provides means both for exact declaration of how the deletion and insertion are effected and for explicit indication that one text is a substitute for another. Further, the exact sequence of readings may also be declared by use of the varSeq attribute on the <rdg> element, as follows:

```
How it
  <app>
     <rdg hand='DHL' varSeq=1><del>galls</del></rdg>
     <rdg hand='DHL' varSeq=2><add>dogs</add> </rdg>
  </app>
me, what a galling shadow
```

Here, the combination of the hand and varSeq attributes suffices to inform the reader of the authorial substitution of 'dogs' for 'galls'.

Similarly, the varSeq attribute might be used in a transcription of the manuscripts of James Joyce's *Ulysses* to indicate the sequence of Joyce's corrections which is implicit in Hans Walther Gabler's reconstruction of the "overlay" levels of Joyce's transcriptions. This third method is the most powerful and unambiguous of the three methods and enables the widest range of processing possibilities. However, it does suffer an apparent disadvantage. It introduces more markup into the text, which can prove a burden to those working without SGML-aware editors. The volume of markup may be reduced by markup minimization, as in the following recoding of the Lawrence example, but some overhead will remain nevertheless:

```
How it
  <app>
     <rdg><del>galls</></>
     <rdg><add>dogs</></>
  </app>
me, what a galling shadow
```

A second disadvantage is that applications of considerable sophistication may be needed to make full use of all the information that may be held within an <app> structure. In the absence of such applications, scholars may feel that the present cost of the more informative coding using <app> structures outweighs the future benefits. In making such decisions, it should however be

kept in mind that the capabilities of software at the time a project begins will often be wholly irrelevant when the project is completed some years later.

The Lawrence example above shows the three methods used for encoding a single substitution of one reading for another. The same three methods may also be used to encode longer sequences of substitutions. In the example from William James, first written out by James as "One must have lived longer with this system, to appreciate its advantages" the word 'this' is first replaced by 'such a' and this is then replaced by 'a'.[8] This may be encoded using the first method, with the sequence of substitutions shown by the nesting of <corr> elements:

```
One must have lived longer with
<corr sic='this'><corr sic='such a'>a</corr></corr>
system, to appreciate its advantages.
```

It may be encoded using the second method, with the two changes being treated as a sequence of additions and deletions:

```
One must have lived longer with
<del>this</del>
<del><add>such a</add></del>
<add>a</add> system,
to appreciate its advantages.
```

Note the nesting of an <add> element within a <del> to record text first added, then deleted in the source.

It may be encoded using the third method, with each reading in the series contained in a <rdg> element within an <app> structure:

```
One must have lived longer with
  <app>
     <rdg varSeq=1><del>this</del>
     <rdg varSeq=2><del><add>such a</add></del>
     <rdg varSeq=3><add>a</add>
  </app>
system, to appreciate its advantages.
```

The three encodings of this slightly more complex example illustrate the general truth that the more information involving substitutions there is to be encoded, the clearer become the advantages of the use of the <app> method over the other two methods. As a rule, it is recommended that the <app> method be used for encoding substitutions of any complexity. It is also desirable that the one method be used throughout any one transcription. Accordingly, the <app> method is recommended for text critical transcription of primary textual materials requiring encoding of instances of other than straightforward substitution.

18.1.6 Cancellation of Deletions and Other Markings

An author or scribe may mark a word or phrase in some way, and then on reflection decide to cancel the marking. For example, text may be marked for deletion and the deletion then cancelled, thus restoring the deleted text. Such cancellation may be indicated by the <restore> element:

<restore> indicates restoration of text to an earlier state by cancellation of an editorial or authorial marking or instruction. Attributes include:

- **type** indicates the action cancelled by the restoration.
- **desc** gives a prose description of the means of restoration.
- **resp** signifies the editor or transcriber responsible for identifying the hand of the restoration.
- **cert** signifies the degree of certainty ascribed to the identification of the hand of the restoration.
- **hand** signifies the hand of the agent which made the restoration.

Presume that Lawrence decided to restore 'my' to the phrase of *Eloi, Eloi, lama sabachthani* first written "For I hate this my body", with the 'my' first deleted then restored by writing "stet" in the margin. This may be encoded:

[8] The manuscript contains several other substitutions, ignored here for the sake of clarity.

```
For I hate this
<restore hand='DHL' desc='marginal "stet"'><del>my</del></restore>
body
```

The <restore> element is defined as follows:

```
<!-- 18.1.6:  Cancelled Deletions                                  -->
<!ELEMENT restore       - O  (%phrase.seq;)                        >
<!ATTLIST restore            %a.global;
          wit                CDATA               #IMPLIED
          cause              CDATA               #IMPLIED
          varSeq             NUMBER              #IMPLIED
          type               CDATA               #IMPLIED
          desc               CDATA               #IMPLIED
          resp               IDREF               %INHERITED
          cert               CDATA               #IMPLIED
          hand               IDREF               %INHERITED     >
<!-- This fragment is used in sec. 18                            -->
```

18.1.7 Text Omitted from or Supplied in the Transcription

Where text is not transcribed, whether because of damage to the original, or because it is illegible, or because of editorial policy, the <gap> core element should be used to register the omission; where text not present in the source is supplied (whether conjecturally or from other witnesses) to fill an apparent gap in the text, it should be marked using the <supplied> element provided by the tag set defined in this chapter.

<gap> indicates a point where material has been omitted in a transcription, whether for editorial reasons described in the TEI header, as part of sampling practice, or because the material is illegible or inaudible. Attributes include:

- **desc** gives a description of the omitted text.
- **reason** gives the reason for omission. Sample values include "sampling", "illegible", "inaudible", "irrelevant", "canceled", "canceled and illegible".
- **extent** indicates approximately how much text has been omitted from the transcription, in letters, minims, inches, or any appropriate unit, either because of editorial policy or because a deletion, damage or other cause has rendered transcription impossible.
- **resp** indicates the editor, transcriber or encoder responsible for the decision not to provide any transcription of the text and hence the application of the <gap> tag.
- **hand** In the case of text omitted from the transcription because of deliberate deletion by an identifiable hand, signifies the hand which made the deletion.
- **agent** In the case of text omitted from the transcription because of damage or other phenomenon resulting from an identifiable cause, signifies the causative agent.

<supplied> signifies text supplied by the transcriber or editor in place of text which cannot be read, either because of physical damage or loss in the original or because it is illegible for any reason. Attributes include:

- **reason** indicates why the text has had to be supplied.
- **resp** indicates the individual responsible for supplying the letter, word or passage contained within the <supplied> element.
- **hand** Where the presumed loss of text leading to the supplying of text arises from action (partial deletion, etc.) assignable to an identifiable hand, signifies the hand responsible for the action.
- **agent** where the presumed loss of text leading to the supplying of text arises from an identifiable cause, signifies the causative agent.
- **source** states the source of the supplied text.

By its nature, the <gap> element must have no content. It should be used wherever an authorial or scribal erasure is so successful, or the text is so illegible, that nothing can be read. In the Beerbohm manuscript of *The Golden Drugget* cited above, for example, the author has erased several passages by inking them over completely:

```
Others <gap reason='cancelled' hand='MB' extent='10cm'>--and
here is one of them...
```

In an autograph letter of Sydney Smith in the Pierpont Morgan library (Klinkenborg 11), three words in the signature are quite illegible:

```
I am dr Sr yr <gap reason='illegible'
                   hand='SS'
                   extent='3 words'>Sydney Smith
```

It is possible, but not always necessary, to provide measurements precise to the millimeter or even to the printer's point. The degree of precision attempted will vary with the purpose of the encoding and the nature of the material.

In cases where there is damage, or a degree of illegibility, but the text is nevertheless legible and is transcribed, the <gap> element should not be used. Instead, the passage should be marked using one or more of the elements <damage> and <unclear>, which are described in section 18.2.3 ('Damage, Illegibility, and Supplied Text') on p. 460.

If the source text is completely illegible or missing, and new text is supplied to fill the gap, it should be marked as <supplied>. If another (imaginary) copy of the letter above preserved the signature as reading "I am dear Sir your very humble Servt Sydney Smith", the text illegible in the autograph might be supplied in the transcription:

```
I am dr Sr yr
<supplied reason='illegible'
          resp='RW'
          source='amanuensis copy'>very humble Servt</>
Sydney Smith
```

Both <gap> and <supplied> may be used in combination with <unclear>, <damage>, and other elements; for discussion, see section 18.2.4 ('The Use of the Gap, Del, Damage, Unclear and Supplied Tags in Combination') on p. 462.

As noted, <gap> is defined in the core tag set. The <supplied> element is declared thus:

```
<!-- 18.1.7:  Supplied Text                                    -->
<!ELEMENT supplied      - O  (%paraContent;)                     >
<!ATTLIST supplied           %a.global;
          reason             CDATA               #IMPLIED
          resp               CDATA               %INHERITED
          hand               IDREF               %INHERITED
          agent              CDATA               #IMPLIED
          source             CDATA               #IMPLIED       >
<!-- This fragment is used in sec. 18                          -->
```

18.2 Non-Linguistic Phenomena in the Source

This section describes methods for recording a number of non-linguistic characteristics of the source text which are often of particular interest in the transcription of primary sources: points at which one scribe takes over from another, or at which ink, pen, or other characteristics of the writing change; points at which the source is damaged or imperfectly legible; and unusual spaces or lines in the source. A discussion of the usage of the hand, resp, and cert attributes is also included. Methods for recording page breaks, column breaks, and line breaks in the source are described in section 6.6 ('Simple Links and Cross References') on p. 147.

18.2.1 Document Hands

For many text-critical purposes it is important to signal the person responsible (the "hand") for the writing of a whole document, a stretch of text within a document, or a particular feature within the document. The hand may be of a known and named scribe or author, as "DHL", or may be described by an anonymous formula, as "hand one". Where the hand is associated with a particular feature tagged within a document, this may be indicated by the value of the hand

attribute on that feature. The examples given above of the use of the hand attribute with coding of additions and deletions illustrate this.

In other cases, it may be necessary to identify a document hand without there being any association of that hand with any specific tagged document feature. The <handList> and <hand> elements are used in the TEI header (in the <profileDesc> element) to define each unique hand or scribe distinguished by the encoder in the document. One such element must appear within the header for each hand distinguished in the text. Each location where a change of hands occurs may then be marked in the text by the empty <handShift> element.

<hand> used in the header to define each distinct scribe or handwriting style. Attributes include:

- **hand** unique identifier, either numeric or alphanumeric, used thereafter in the document to refer to this scribe or handwriting style.
- **scribe** gives the name of, or other identifier for, the scribe.
- **style** indicates recognized writing styles.
- **lang** indicates dominant language of hand.
- **ink** describes colour of ink, e.g. 'brown'. May also be used to indicate the writing medium, e.g. 'pencil',
- **character** used to describe other characteristics of the hand, particularly those related to the quality of the writing.
- **first** indicates the first scribe in the document.
- **resp** signifies the editor or transcriber responsible for identifying the hand.

<handList> contains a series of <hand> elements listing the different hands of the source.

<handShift> marks the beginning of a sequence of text written in a new hand, or of a change in the scribe, writing style, ink or character of the document hand. Attributes include:

- **new** identifies the new hand.
- **old** identifies the old hand.
- **style** indicates recognised writing styles
- **ink** describes colour of ink, e.g. 'brown'. May also be used to indicate the writing medium, e.g. 'pencil'
- **character** used to describe other characteristics of the hand, particularly those related to the quality of the writing.
- **resp** signifies the editor or transcriber responsible for identifying the change of hand.

The attributes old and new on the <handShift> element refer to the order of the text in the transcription: "old" is the material before the <handShift>, "new" the material following. This will ordinarily, but not necessarily, be the order in which the material was originally written. Neither attribute is required but both are recommended where there is a new hand, as opposed to a new writing style in the one hand. The character attribute will be most often used to encode descriptive shifts which the transcriber perceives within a manuscript and which may or may not be associated with or denote changes in scribe or content. The particular values encoded will depend upon the needs of the transcriber. Where many values are to be encoded, feature structures provide an alternative means of encoding these.

A single hand may employ different writing styles and inks within a document, or may change character. For example, the writing style might shift from "anglicana" to "secretary", or the ink from blue to brown, or the character of the hand may change. Any such changes should be indicated by assigning a new value to the appropriate attribute within the <handShift> element. The one hand may employ different renditions within the one writing style, for example medieval scribes indicating a structural division by emboldening all the words within a line. These should be indicated by use of the rend attribute on an element, in the same manner as underlining, emboldening, font shifts, etc., in transcription of a printed text, rather than by introducing a new <handShift> element.

In this example[9] first the document hands are declared in the header:

[9] From the Wiltshire Record Office, Dean of Sarum Churchwardens' presentments, 1731, Hurst; the transcription was provided by Donald A. Spaeth.

```
<teiHeader>
<!-- ... -->
<profileDesc>
   <!-- ... -->
   <handList>
      <hand id=h1 type='copperplate'
                  ink='brown'
                  character='regular'
                  first='yes'
                  resp='das'>
      <hand id=h2 type='none'
                  colour='brown'
                  character='unschooled'
                  resp='das'>
   </handList>
   <!-- ... -->
</profileDesc>
</teiHeader>
```

Then the change of hand is indicated in the text:

```
<!-- ... -->
and that good Order Decency and regular worship
may be once more introduced and Established in this
Parish according to the Rules and Ceremonies of the
Church of England and as under a good Consciencious
and sober Curate there would and ought to be
<handShift new='h2' old='h1' resp='das'>
and for that purpose the parishioners pray
```

In this example[10] there is a change of ink within the one hand. This is indicated by a new value for the ink attribute on the <handShift> element:

```
<l>When wolde the cat dwelle in his ynne</l>
<handShift ink=black>
<l>And if the cattes skynne be slyk and gaye</l>
```

These elements are declared as follows:

```
<!-- 18.2.1:  Hand Shifts                                    -->
<!ELEMENT hand          - O  EMPTY                            >
<!ATTLIST hand
          id                 ID                  #IMPLIED
          n                  CDATA               #IMPLIED
          rend               CDATA               #IMPLIED
          hand               CDATA               #REQUIRED
          scribe             CDATA               #IMPLIED
          style              CDATA               #IMPLIED
          lang               CDATA               #IMPLIED
          ink                CDATA               #IMPLIED
          character          CDATA               #IMPLIED
          first              CDATA               #IMPLIED
          resp               CDATA               %INHERITED    >
<!ELEMENT handShift     - O  EMPTY                            >
<!ATTLIST handShift          %a.global;
          new                IDREF               #IMPLIED
          old                IDREF               #IMPLIED
          style              CDATA               #IMPLIED
          ink                CDATA               #IMPLIED
          character          CDATA               #IMPLIED
          resp               IDREF               %INHERITED    >
<!ELEMENT handList      - O  (hand*)                          >
```

[10] From folio 52 recto of the Holkham manuscript of Chaucer's *Canterbury Tales*.

```
<!ATTLIST handList          %a.global;                        >
<!-- This fragment is used in sec. 18                       -->
```

18.2.2 Hand, Responsibility, and Certainty Attributes

The hand and resp attributes have similar, but not identical, meanings. Observe their distinctive uses in the following encoding of the William James passage mentioned above in section 18.1.3 ('Correction and Conjecture') on p. 447. In this example, the 'But' inserted by James is tagged as an <add>, and the consequent editorial correction of 'One' to 'one' treated separately:

```
<add loc='sup' resp=FB hand=WJ>But</add>
<corr sic='One' resp=FB>one</> must have lived ...
```

As in this example, hand should be reserved for indicating the hand of any form of marking—here, addition but also deletion, correction, annotation, underlining, etc.—within the primary text being transcribed. The scribal or authorial responsibility for this marking may be inferred from the value of the hand attribute. The value of the hand attribute should be one of the hand identifiers declared in the document header (see section 18.2.1 ('Document Hands') on p. 456).

As in this example, the resp on a particular element should be used only to indicate the particular aspect of responsibility defined in these *Guidelines* as appropriate to the resp attribute for that element. In the case of the <add> element, the resp attribute is defined as signifying the responsibility for identifying the hand of the addition: here, Bowers' identification of the hand as that of William James. In the case of the <corr> element, the resp attribute is defined as signifying the responsibility for supplying the intellectual content of the correction reported in the transcription: here, Bowers' correction of "One" to "one".

As these examples show, the field of application of the resp attributes varies from element to element. In some cases, it applies to the content of the element (<corr> and <expan>); in others it applies to the value of a particular attribute (<sic>, <abbr>, <del>, etc.). In all cases where both the cert and resp attributes are defined for a particular element, the two attributes refer to the same aspect of the markup. The one indicates who is intellectually responsible for some item of information, the other indicates the degree of confidence in the information. Thus, for a correction, the resp attribute signifies the person responsible for supplying the correction, while the cert attribute signifies the degree of editorial confidence felt in that correction. For the expansion of an abbreviation, the resp attribute signifies the person responsible for supplying the expansion and the cert attribute signifies the degree of editorial confidence felt in the expansion.

This close definition of the use of the resp and cert attributes with each element is intended to provide for the most frequent circumstances in which encoders might wish to make unambiguous statements regarding the responsibility for and certainty of aspects of their encoding. The resp and cert attributes, as so defined, give a convenient mechanism for this. However, there will be cases where it is desired to state responsibility for and certainty concerning other aspects of the encoding. For example, one may wish in the case of an apparent addition to state the responsibility for the use of the <add> element, rather than the responsibility for identifying the hand of the addition. It may also be that one editor may make an electronic transcription of another editor's printed transcription of a manuscript text — here, one will wish to assign layers of responsibility, so as to allow the reader to determine exactly what in the final machine-readable transcription was the responsibility of each editor. In these complex cases of divided editorial responsibility for and certainty concerning the content, attributes and application of a particular element, the more general mechanisms for representing certainty and responsibility described in chapter 17 ('Certainty and Responsibility') on p. 435 should be used.

The fields of reference of the resp and cert attributes for each element have been chosen to enable what are felt as the most frequent likely statements an encoder may wish to make concerning the areas of responsibility and certainty related to that element. It is open to each local transcription scheme to vary the use of the resp and cert attributes on particular elements where it is felt convenient. This practice should be documented in the <encodingDesc> element in the file header. Further, it is recommended that before interchange any such local usage of these attributes be converted to conformancy with the definitions of the resp and cert attributes given in these *Guidelines*. Use of the resp and cert in interchange documents in ways not here defined may lead to unpredictable results.

It should be noted that the certainty and responsibility mechanisms described in chapter 17 ('Certainty and Responsibility') on p. 435 replicate all the functions of the resp and cert attributes on particular elements. For example, the encoding of Donaldson's conjectured emendation of 'wight' to 'wright' in line 117 of Chaucer's *Wife of Bath's Prologue*, (see 18.1.3 ('Correction and Conjecture') on p. 447) may be encoded as follows using the resp and cert attributes on the <corr> element:

```
<corr sic="wight" resp=ETD cert=70>wright</>
```

Exactly the same information could be conveyed using the certainty and responsibility mechanisms, as follows:

```
<corr sic="wight" id=c117>wright</>

<!-- ... certainty and responsibility elements may be elsewhere -->
<certainty target=c117 locus='#gicontent' degree=70>
<respons   target=c117 locus='#gicontent' resp=ETD>
```

The choice of which mechanism to use is left to the encoder. In transcriptions where only such statements of responsibility and certainty are made as can be accommodated within the resp and cert attributes of particular elements, it will be economical to use the resp and cert attributes of those elements. Where many statements of responsibility and certainty are made which cannot be so accommodated, it may be economical to use the <respons> and <certainty> elements throughout.

The above discussion supposes that in each case an encoder is able to specify exactly what it is that one wishes to state responsibility for and certainty about. Situations may arise when an encoder wishes to make a statement concerning certainty or responsibility but is unable or unwilling to specify so precisely the domain of the certainty or responsibility. In these cases, the <note> element may be used with the type attribute set to "cert" or "resp" and the content of the note giving a prose description of the state of affairs.

18.2.3 Damage, Illegibility, and Supplied Text

The <gap> and <supplied> elements described above (section 18.1.7 ('Text Omitted from or Supplied in the Transcription') on p. 455) should be used with appropriate attributes where the degree of damage or illegibility in a text is such that nothing can be read and the text must be either omitted or supplied either conjecturally or from other sources. In many cases, however, despite damage or illegibility, the text may yet be read with reasonable confidence. In these cases, the following elements should be used:

<damage> contains an area of damage to the text witness. Attributes include:

- **type** classifies the damage according to any convenient typology.
- **resp** indicates the individual responsible for identifying the area of damage.
- **hand** In the case of damage (deliberate defacement, etc.) assignable to an identifiable hand, signifies the hand responsible for the damage.
- **agent** In the case of damage resulting from an identifiable cause, signifies the causative agent.
- **degree** Signifies the degree of damage according to a convenient scale. The <damage> tag with the degree attribute should only be used where the text may be read with some confidence; text supplied from other sources should be tagged as <supplied>.
- **extent** indicates approximately how much text is in the damaged area, in letters, minims, inches, or any appropriate unit, where this cannot be deduced from the contents of the tag. For example, the damage may span structural divisions in the text so that the tag must then be empty of content.

<unclear> contains a word, phrase, or passage which cannot be transcribed with certainty because it is illegible or inaudible in the source. Attributes include:

- **reason** indicates why the material is hard to transcribe.
- **resp** indicates the individual responsible for the transcription of the letter, word or passage contained with the <unclear> element.
- **cert** signifies the degree of certainty ascribed to the transcription of the text contained within the <unclear> element.

hand Where the difficulty in transcription arises from action (partial deletion, etc.) assignable to an identifiable hand, signifies the hand responsible for the action.

agent Where the difficulty in transcription arises from an identifiable cause, signifies the causative agent.

The following examples refer to the recto of folio 5 of the unique manuscript of the Elder Edda.[11] Here, the manuscript of *Voluspá* has been damaged through irregular rubbing so that letters in various places are obscured and in some cases cannot be read at all. The existence of the damage may be registered in general for this leaf by use of the <damage> element.

```
<damage cause='rubbing at edges' extent='whole leaf'> ... </>
```

However, in fact the damage crosses structural divisions, so the <damage> element does not nest properly within the containing elements. The simplest method to solve this problem is to split the element into two fragments, one within each structural division:

```
<div>
  <!-- beginning of division ... -->
  <!-- page break, beginning of damage -->
  <pb n='5r'>
  <damage cause='rubbing at edges' extent='whole leaf'>
  <!-- text continues -->
  </damage>
</div>
<div>
  <damage cause='rubbing at edges, continued' extent='whole leaf'>
  <!-- beginning of new text division ... -->
  <!-- page break, end of this damaged section -->
  </damage>
  <pb n='5v'>
  <!-- text continues ... -->
</div>
```

For other techniques of handling non-nesting information, see chapter 31 ('Multiple Hierarchies') on p. 633.

In the first line of this leaf, the transcriber may believe that the last three letters of 'daga' can be read clearly despite the damage:

```
um aldr d<damage>aga</> yndisniota
```

Alternatively, the letters in question may be only imperfectly legible on account of the damage; this state of affairs may be indicated by nesting an <unclear> element within the <damage> element.

```
um aldr d<damage><unclear>aga</></> yndisniota
```

Alternatively, the transcriber may not feel able to read the last three letters of 'daga' but may wish to supply them by conjecture. Note the use of the source attribute to assign the conjecture to Finnur JJónsson:

```
um aldr d<supplied reason='rubbing'
                   source=FJ>aga</supplied>
yndisniota
```

The <supplied> element may if desired be enclosed within a <damage> element:

```
um aldr d<damage cause='rubbing'>
<supplied source=FJ>aga</supplied></damage>
yndisniota
```

Contrast the use of <gap> in the next line, where the transcriber believes that four letters cannot be read at all because of the damage:

```
&Thorn;ar k&hook-o;mr inn dimmi dreki fliugandi
na&thorn;r frann ne&thorn;an
<gap reason='rubbing' extent='4'>
```

[11] Codex Regius, ed. L. F. A. Wimmer and F. Jónsson (Copenhagen 1891).

As with <supplied>, this <gap> might be enclosed by a <damage> element.

In these examples, various phenomena of illegibility and conjecture all result from the one cause, an area of damage to the text — rubbing at various points — which is not continuous in the text, affecting it at irregular points. In these cases, the <join> element may be used to indicate which tagged features are part of the same physical phenomenon. (See chapter 14 ('Linking, Segmentation, and Alignment') on p. 331 for more details.)

The above examples record imperfect legibility due to damage. When imperfect legibility is due to some other reason (typically because the handwriting is ill-formed), the <unclear> element should be used without any enclosing <damage> element. In Robert Southey's autograph of *The Life of Cowper*,[12] the final six letters of 'attention' are difficult to read because of the haste of the writing, though reasonably certain from the context.

```
and from time to time invited in like manner his
att<unclear>ention</unclear>
```

The cert attribute on the <unclear> element may be used to indicate the level of editorial confidence in the reading contained within it.

The <damage> element is defined formally as follows:

```
<!-- 18.2.3:  Damage and Illegiblity                          -->
<!ELEMENT damage        - O  (%paraContent;)                    >
<!ATTLIST damage             %a.global;
          type               CDATA              #IMPLIED
          extent             CDATA              #IMPLIED
          resp               IDREF              %INHERITED
          hand               IDREF              %INHERITED
          agent              CDATA              #IMPLIED
          degree             CDATA              #IMPLIED       >
<!-- This fragment is used in sec. 18                         -->
```

The <unclear> element is defined in section 6.5 ('Simple Editorial Changes') on p. 140.

18.2.4 The Use of the Gap, Del, Damage, Unclear and Supplied Tags in Combination

The <gap>, <damage>, <unclear>, <supplied> and <del> elements may be closely allied in their use. For example, an area of damage in a primary source might be encoded with any one of the first four of these elements, depending on how far the damage has affected the readability of the text. Further, certain of the elements may nest within one another. The examples given in the last sections illustrate something of how these elements are to be distinguished in use. This may be formulated as follows:

- where the text has been rendered completely illegible by deletion or damage and no text is supplied by the editor in place of what is lost: place an empty <gap> element at the point of deletion or damage. Use the reason attribute to state the cause (damage, deletion, etc.) of the loss of text.
- where the text has been rendered completely illegible by deletion or damage and text is supplied by the editor in place of what is lost: surround the text supplied at the point of deletion or damage with the <supplied> element. Use the reason attribute to state the cause (damage, deletion, etc.) of the loss of text leading to the need to supply the text.
- where the text has been rendered partly illegible by deletion or damage so that the text can be read but without perfect confidence: transcribe the text and surround it with the <unclear> element. Use the reason attribute to state the cause (damage, deletion, etc.) of the uncertainty in transcription and the cert attribute to indicate the confidence in the transcription.
- where there is deletion or damage but the text can be read with perfect confidence: transcribe the text and surround it with the <del> element (for deletion) or the <damage> element (for damage). Use appropriate attribute values to indicate the cause and type of deletion or damage. Observe that the degree attribute on the <damage> element permits the encoding

[12] In Pierpont Morgan MA 412, (Klinkenborg 15).

to show that a letter, word or phrase is not perfectly preserved, though it may be read with confidence.

- where there is an area of deletion or damage and parts of the text within that area can be read with perfect confidence, other parts with less confidence, other parts not at all: in transcription, surround the whole area with the <del> element (for deletion; or the <delSpan> element where it crosses a structural boundary); or the <damage> element (for damage). Text within the damaged area which can be read with perfect confidence needs no further tagging. Text within the damaged area which can not be read with perfect confidence may be surrounded with the <unclear> element. Places within the damaged area where the text has been rendered completedly illegible and no text is supplied by the editor may be marked with the <gap> element. For each element, one may use appropriate attribute values to indicate the cause and type of deletion or damage and the certainty of the reading.

The rules for combinations of the <add> and <del> elements, and for the interpretation of such combinations, are similar:

- when one addition (<add id=a1>) includes another (<add id=a2>), it indicates that an addition (a1) was first made to the text, and later a second addition (a2) was made to the text already added:

```
This is the text
<add id=a1>with some added
   <add id=a2>(interlinear!)</add>
material</add>
as written.
```

- when one deletion (<del id=d1>) nests within another (<del id=d2>), it indicates that the author wrote a passage, deleted part of it (d1), and then later deleted the entire passage (d2).

```
<del id=d2>This sentence contains
some <del id=d1>redundant</del> unnecessary
verbiage.</del>
```

- when an addition nests within a deletion, the normal interpretation will be that an addition was made within a passage later deleted in its entirety.
- when a deletion nests within an addition, it indicates that a deletion was made within a passage earlier added.

18.2.5 Space

The presence of significant space in the text being transcribed may be indicated by the <space> element. The author or scribe may have left space for a word, or for an initial capital, and for some reason the word or capital was never supplied and the space left empty. This element should not be used to mark normal inter-word space or the like.

<space> indicates the location of a significant space in the copy text. Attributes include:

dim indicates whether the space is horizontal or vertical. Legal values are:

horizontal the space is horizontal.

vertical the space is vertical.

extent indicates approximately how large the space is, in letters, minims, inches, or other appropriate unit.

resp incidates the individual responsible for identifying and measuring the space.

In line 694 of Chaucer's *Wife of Bath's Prologue* in the Holkham manuscript the scribe has left a space for a word where other manuscripts read 'preestes':

```
By god if wommen had writen storyes
As <space extent='7'> han within her oratoryes
```

The <supplied> element discussed in the previous section may be used to supply the text presumed missing:

```
By god if wommen had writen storyes
As <supplied reason='space' resp='ES' source='Hg'>
preestes</supplied> han within her oratoryes
```

Here, the fact of the space within the manuscript is indicated by the value of the reason attribute. The source of the supplied text is shown by the value of the source attribute as the Hengwrt manuscript; the transcriber responsible for supplying the text is ES.

The <space> element is formally defined thus:

```
<!-- 18.2.5:  Spaces in the source                          -->
<!ELEMENT space         - O  EMPTY                          >
<!ATTLIST space              %a.global;
          dim                (horizontal | vertical)
                                                 #IMPLIED
          extent             CDATA               #IMPLIED
          resp               CDATA               #IMPLIED    >
<!-- This fragment is used in sec. 18                       -->
```

18.2.6 Lines

The most common form of marking of text in manuscripts is by lines written under, beside or through the text. The lines themselves may be of various types: they may be solid, dashed, or dotted, doubled or tripled, wavy or straight, or a combination of these and other renderings. The line may be used for emphasis, or to mark a foreign or technical term, or to signal a quotation or a title, etc.: the elements <emph>, <foreign>, <term>, <mentioned>, <title> may be used for these. Frequently, a scholar may judge that a line is used to delete text: the <del> element is available to indicate this. In all these cases, the rend attribute may be used on these or other elements to indicate that the text is marked by a line and the style of the line. Thus, Lawrence's deletion by strike-through of 'my' in the autograph of *Eloi, Eloi, lama sabachthani* is noted:

```
For I hate this
<del hand='DHL' rend='strikethrough'>my</del> body,
which is so dear to me
```

There will be instances, however, where a scholar wishes only to register the occurrence of lines in the text, without making any judgement as to what the lines signify. In these the <hi> element may be used, with the rend attribute to mark the style of line. In the manuscript of a letter by Robert Browning to George Moulton-Barrett (Pierpont Morgan MA 310, Klinkenborg 23), the underlining of the phrase "had obtained all the letters to Mr Boyd" may be marked:

```
I have once,--by declaring I would prosecute
by law--, hindered a man's proceedings who
<hi rend=underline>had obtained all the letters
to Mr Boyd</hi>
```

The above examples presume the common case where a single word or phrase is marked by a line, with no doubt as to where the marking begins or ends and with no overlapping of the area of text with other marked areas of text. Where there is doubt, the <certainty> element may be used to record the doubt. In the Browning example cited above the underlining actually begins half-way under 'who', and this uncertainty could be remarked as follows:

```
I have once,--by declaring I would prosecute
by law--, hindered a man's proceedings who
<hi id=cstart1 rend=underline>had obtained all
the letters to Mr Boyd</hi>

<!-- ... -->
<certainty target=cstart1
           locus='#startloc'
           desc='may begin with previous word'
           degree='0.70'>
```

Where the area of text marked overlaps other areas of text, for example crossing a structural division, one of the span mechanisms outlined in these Guidelines may be used. Where the line is thought to mark a deletion, the <delSpan> element may be used. Where it is desired simply to record the marking of a span of text in circumstances where it is not possible to surround the

text with a <hi> element, the element may be used with the rend attribute indicating the style of line-marking.

More work needs to be done on clarifying the treatment of other textual features marked by lines which might so overlap or nest. For example, in many Middle English manuscripts (e.g. the Jesus and Digby verse collections) marginal sidebars may indicate metrical structure: couplets may be linked in pairs, with the pairs themselves linked into stanzas. Or, marginal sidebars may indicate emphasis, or may point out a region of text on which there is some annotation: in many manuscripts of Chaucer's *Wife of Bath's Prologue* lines 655-8 are marked with nesting parentheses against which the scribe has written 'nota'.

At the lowest level, all such features could be captured by use of the <note> element, containing a prose description of the manuscript at this point. It is not yet clear how best to mark up such phenomena so as to obtain more usefully structured encodings. For example, in the Chaucer example just cited, one may wish to record that the 'nota' is written in the Hengwrt manuscript in the right margin against a single large left parenthesis bracketing the four lines, with two right parentheses in the right margin bracketing two overlapping pairs of lines: the first and third, the second and fourth. The <note> element allows us to record that the scribe wrote 'nota', but is not well-adapted to show that the 'nota' points both at all four lines and at two pairs of lines within the four lines. Work will continue in this area.

18.3 Headers, Footers, and Similar Matter

As a rule, matter associated with the page break (signature, catchword, page number) should be drawn into the <pb> element as attributes: see section 6.9 ('Reference Systems') on p. 155. In text-critical situations where these elements need tagging in their own right (for instance, when the catch-word presents a variant reading, or spacing in the header or footer is significant for compositor identification) the element <fw> may be used:

<fw> contains a running head (e.g. a "header", "footer"), catchword, or similar material appearing on the current page. Attributes include:

place indicates where on the page this material appears. Suggested values include:

top top of the page.

bot bottom of the page.

left in left margin.

right in right margin.

The name 'fw' is short for "forme work". It may be used to encode any of the unchanging portions of a page forme, such as:

- running heads (whether repeated on every page, or changing on every page)
- running footers
- page numbers
- catch-words
- other material repeated from page to page, which falls outside the stream of the text

It should not be used for marginal glosses, annotations, or textual variants, which should be tagged using <gloss>, <note>, or the text-critical tags described in chapter 19 ('Critical Apparatus') on p. 467, respectively.

For example:

```
<fw place=top-centre type=head>Po&euml;ms.</fw>
<fw place=top-right type=pageno>29</fw>
<fw place=bot-centre type=sig>E3</fw>
<fw place=bot-right type=catch>TEMPLE</fw>
```

The formal declaration for the <fw> element is this:

```
<!-- 18.3:  Headers and footers                                  -->
<!ELEMENT fw            - O  (%phrase.seq;)                      >
<!ATTLIST fw                 %a.global;
```

```
          type              CDATA              #IMPLIED
          place             CDATA              #IMPLIED       >
<!-- This fragment is used in sec. 18                          -->
```

18.4 Other Primary Source Features not Covered in These Guidelines

We repeat the advice given at the beginning of this chapter, that these recommendations are not intended to meet every transcriptional circumstance ever likely to be faced by any scholar. They are intended rather as a base to enable encoding of the most common phenomena found in the course of scholarly transcription of primary source materials. These guidelines particularly do not address the encoding of physical description of textual witnesses: the materials of the carrier, the medium of the inscribing implement, the layout of the inscription upon the material, the organisation of the carrier materials themselves (as quiring, collation, etc.), authorial instructions or scribal markup, etc. Some of these issues may be covered in future editions of these guidelines.

Chapter 19

Critical Apparatus

Scholarly editions of texts, especially texts of great antiquity or importance, often record some or all of the known variations among different *witnesses* to the text. Witnesses to a text may include authorial or other manuscripts, printed editions of the work, early translations, or quotations of a work in other texts. Information concerning variant readings of a text may be accumulated in highly structured form in a critical apparatus of variants. This chapter defines an additional tag set for use in encoding such an apparatus of variants, which may be used in conjunction with any of the base tag sets defined in these Guidelines. It also defines an element class which provides extra attributes for some elements of the core tag set when this additional tag set is selected.

This tag set is selected as described in 3.3 ('Invocation of the TEI DTD') on p. 39; in a document which uses the markup described in this chapter, the document type declaration should contain the following declaration of the entity *TEI.textcrit*, or an equivalent one:

```
<!ENTITY % TEI.textcrit 'INCLUDE'>
```

The entire document type declaration for a document using this additional tag set together with the base tag set for prose might look like this:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
   <!ENTITY TEI.prose     'INCLUDE' >
   <!ENTITY TEI.textcrit  'INCLUDE' >
]>
```

The overall document type declaration for this additional tag set has the following structure. First, the file *teitc2.ent* defines some element classes relevant to this tag set:

```
<!-- 19:  Entity classes for text criticism                  -->
<!-- Text Encoding Initiative: Guidelines for Electronic     -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.   -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy   -->
<!-- in any form is granted, provided this notice is         -->
<!-- included in all copies.                                 -->

<!-- These materials may not be altered; modifications to    -->
<!-- these DTDs should be performed as specified in the      -->
<!-- Guidelines in chapter "Modifying the TEI DTD."          -->

<!-- These materials subject to revision. Current versions   -->
<!-- are available from the Text Encoding Initiative.        -->

<!ENTITY % x.fragmentary ''                                   >
<!ENTITY % m.fragmentary '%x.fragmentary lacunaEnd |
          lacunaStart | witEnd | witStart'                    >
<!ENTITY % a.fragmentary '
         wit               CDATA              #IMPLIED'     >
<!ENTITY % a.readings '
         wit               CDATA              #IMPLIED
```

```
          type              CDATA             #IMPLIED
          cause             CDATA             #IMPLIED
          varSeq            NUMBER            #IMPLIED
          resp              CDATA             %INHERITED
          hand              IDREF             %INHERITED'    >
```

The file *teitc2.dtd* defines the elements themselves:

```
<!-- 19:  Tags for text criticism                                  -->
<!-- Text Encoding Initiative: Guidelines for Electronic       -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.     -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy     -->
<!-- in any form is granted, provided this notice is           -->
<!-- included in all copies.                                   -->

<!-- These materials may not be altered; modifications to      -->
<!-- these DTDs should be performed as specified in the        -->
<!-- Guidelines in chapter "Modifying the TEI DTD."            -->

<!-- These materials subject to revision. Current versions     -->
<!-- are available from the Text Encoding Initiative.          -->

<!-- ... declarations from section 19.1.1                      -->
<!--     (Apparatus entry)                                     -->
<!--     go here ...                                           -->
<!-- ... declarations from section 19.1.2                      -->
<!--     (Readings)                                            -->
<!--     go here ...                                           -->
<!-- ... declarations from section 19.1.3                      -->
<!--     (Reading Groups)                                      -->
<!--     go here ...                                           -->
<!-- ... declarations from section 19.1.4.1                    -->
<!--     (Witness Details)                                     -->
<!--     go here ...                                           -->
<!-- ... declarations from section 19.1.4.2                    -->
<!--     (Source-text Witness Lists in Apparatus)              -->
<!--     go here ...                                           -->
<!-- ... declarations from section 19.1.4.3                    -->
<!--     (Witness Lists in Front Matter)                       -->
<!--     go here ...                                           -->
<!-- ... declarations from section 19.1.5                      -->
<!--     (Fragmentary witnesses)                               -->
<!--     go here ...                                           -->
```

Information about variant readings (whether or not represented by a critical apparatus in the source text) may be recorded in a series of *apparatus entries*, each entry documenting one *variation*, or set of readings, in the text. Tags for the apparatus entry and readings, and for the documentation of the witnesses whose readings are included in the apparatus, are described in section 19.1 ('The Apparatus Entry, Readings, and Witnesses') on p. 469. Special tags for fragmentary witnesses are described in section 19.1.5 ('Fragmentary Witnesses') on p. 479. The available methods for embedding the apparatus in the rest of the text, or for linking an external apparatus to the base text, are described in section 19.2 ('Linking the Apparatus to the Text') on p. 480. Finally, several extra attributes for some tags of the core tag set, made available when the additional tag set for text criticism is selected, are documented in section 18.1.1 ('Use of Core Tags for Transcriptional Work') on p. 445.

Many examples given in this chapter refer to the following texts of the opening (usually just line 1) of Chaucer's *Wife of Bath's Prologue*:

E1 Ellesmere, Huntingdon Library 26.C.9: "Experience though noon Auctoritee / Were in this world, were right ynogh to me / To speke of wo that is in mariage; ..."

Hg Hengwrt, National Library of Wales, Aberystwyth, Peniarth 392D: "Experience thogh noon Auctoritee / Were in this world, is right ynogh for me / To speke of wo that is in mariage; ..."

La British Library Lansdowne 851: "Experiment thouh none auctorite / Were in this world, is right ynohe for me / To speke of wo that is in mariage; ..."

Ra2 Bodleian Library Rawlinson Poetic 149: "Eryment though none auctorite / Were in this world, it is right ynow for me / To speke of wo that is in mariage; ..."

19.1 The Apparatus Entry, Readings, and Witnesses

This section introduces the fundamental markup methods used to encode textual variations:

- the <app> element for entries in the critical apparatus: see section 19.1.1 ('The Apparatus Entry') on p. 469.
- elements for identifying individual readings: see section 19.1.2 ('Readings') on p. 470.
- ways of grouping readings together: see section 19.1.3 ('Indicating Subvariation in Apparatus Entries') on p. 472.
- methods of identifying which witnesses support a particular reading, and for describing the witnesses included in the apparatus: see section 19.1.4 ('Witness Information') on p. 475.
- elements for indicating which portions of a text are covered by fragmentary witnesses: see section 19.1.5 ('Fragmentary Witnesses') on p. 479.

19.1.1 The Apparatus Entry

Individual textual variations are encoded using the <app> element, which groups together all the readings constituting the variation. The identification of discrete textual variations or apparatus entries is not a purely mechanical process; different editors may group readings differently. No rules are given here as to how to group readings into apparatus entries; the tags given here may be used to group readings in whatever way the editor finds most perspicuous or useful.

The individual apparatus entry is encoded with the <app> element:

<app> contains one entry in a critical apparatus, with an optional lemma and at least one reading. Attributes include:

- **type** classifies the variation contained in this element according to some convenient typology.
- **from** identifies the beginning of the lemma in the base text, if necessary.
- **to** identifies the endpoint of the lemma in the base text, if necessary.
- **loc** indicates the location of the variation, when the location-referenced method of apparatus markup is used.

The attributes loc, from, and to, are used to link the apparatus entry to the base text. Several methods may be used for such linkage, each involving a slightly different usage for these attributes. Linkage between text and apparatus is described below in section 19.2 ('Linking the Apparatus to the Text') on p. 480.

Each <app> element comprises one or more readings, which in turn are encoded using the <rdg> or other elements, as described in the next section. A very simple partial apparatus for the first line of the *Wife of Bath's Prologue* might take a form something like this:

```
<app>
  <rdg wit='El'>Experience though noon Auctoritee</rdg>
  <rdg wit='La'>Experiment thogh noon Auctoritee</rdg>
  <rdg wit='Ra2'>Eryment though none auctorite</rdg>
</app>
```

Of course, in practice the apparatus will be somewhat more complex. Specifically, it may be desired to record more obviously that manuscripts El and La agree on the words "noon Auctoritee", to indicate a preference for one reading, etc. The following sections on readings, subvariation, and witness information describe some of the more important complications which can arise.

The structure of an <app> element is formally defined as follows:

```
<!-- 19.1.1:  Apparatus entry                                    -->
<!ELEMENT app           - O  (lem?, ((rdg, wit?) | (rdgGrp,
                             wit?))+)                             >
<!ATTLIST app                %a.global;
          type               CDATA              #IMPLIED
          from               IDREF              #IMPLIED
          to                 IDREF              #IMPLIED
          loc                CDATA              #IMPLIED          >
<!-- This fragment is used in sec. 19                            -->
```

19.1.2 Readings

Individual readings are the crucial elements in any critical apparatus of variants. The following elements should be used to tag individual readings within an apparatus entry:

<lem> contains the lemma, or base text, of a textual variation.
<rdg> contains a single reading within a textual variation.

N.B. the term *lemma* is used here in the text-critical sense of 'the reading accepted as that of the original or of the base text' — it is not to be confused with 'the heading of an entry in a reference book, especially a dictionary,' or 'a subsidiary proposition introduced in the proof of some other proposition; a helping theorem.'

In recording readings within an apparatus entry, the <rdg> element may always be used; each <app> must contain at least one <rdg>.

The <lem> element may also be used, under some circumstances, to record the base text of the source edition, to mark the readings of a base witness, to indicate the preference of an editor or encoder for a particular reading, or to make clear, in cases of ambiguity, precisely which portion of the main text the variation applies to. Those who prefer to work without the notion of a base text may prefer not to use it at all. How it is used depends in part on the method chosen for linking the apparatus to the text; for more information, see section 19.2 ('Linking the Apparatus to the Text') on p. 480.

Readings may be encoded individually, or grouped for perspicuity using the <rdgGrp> element described in section 19.1.3 ('Indicating Subvariation in Apparatus Entries') on p. 472.

As members of the attribute class *readings*, both of these elements inherit the following attributes. Some of these attributes are intelligible only if the reading is ascribed to a single witness; others have no such restriction.

wit contains a list of one or more sigla of witnesses attesting a given reading.
type classifies the reading according to some useful typology. Sample values include:

- ***substantive*** the reading offers a substantive variant.
- ***orthographic*** the reading differs only orthographically, not in substance, from other readings.

cause classifies the reading as original or non-original, according to some typology of possible origins.
varSeq provides a number indicating the position of this reading in a sequence, when there is reason to presume a sequence to the variants on any one lemma.
hand signifies the hand responsible for a particular reading in the witness.
resp identifies the editor responsible for asserting a particular reading in the witness.

The wit attribute identifies the witnesses which have the reading in question. It is required if the apparatus gathers together readings from different witnesses, but may be omitted in an apparatus recording the readings of only one witness, e.g. substitutions, divergent opinions on what is in the witness or on how to expand abbreviations, etc. Even in such a one-witness apparatus, however, the wit attribute may still be useful when it is desired to record the occurrence of a particular reading in some other witness. For other methods of identifying the witnesses to a reading, see section 19.1.4 ('Witness Information') on p. 475.

The type attribute allows the encoder to classify readings in any convenient way, for example as substantive variants of the lemma:

```
<app>
<lem wit='El Hg'>Experience</>
<rdg wit='La'  type='substantive'>Experiment</>
<rdg wit='Ra2' type='substantive'>Eryment</>
</app>
```

or as orthographic variants:

```
<app>
  <lem wit='El Ra2'>though</>
  <rdg wit='Hg' type='orthographic'>thogh</>
  <rdg wit='La' type='orthographic'>thouh</>
</app>
```

The varSeq and cause attributes may be used to convey information on the sequence and cause of variation. In the following apparatus fragment, the reading 'Eryment' is tagged as sequential to (derived from) the reading 'Experiment', and the cause is given as loss of the abbreviation for 'per'.

```
<app>
    <rdg varSeq=1 wit='La'>Experiment</>
    <rdg varSeq=2 cause='abbreviation loss'
                  wit='Ra2'>Eryment</>
</app>
```

If a manuscript is written in several hands, and it is desired to report which hand wrote a particular reading, the hand attribute should be used. For example, in the Munich manuscript containing the *Carmina Burana*, the word 'alle' has been changed to 'allen':

```
<l>Swaz hi g&a;t umbe
<l>daz sint alle megede,
<l>die wellent &a;n man
<l><app>
     <rdg wit='M' hand='m1'
                  varSeq=1>alle</>
     <rdg wit='M' hand='m2'
                  varSeq=2
                  cause='nachgetragen'>allen</>
   </app>
   disen sumer g&a;n.

<!-- entity &a; is used here for &acirc; -->
```

Similarly, if a witness is hard to decipher, it may be desired to indicate responsibility for the claim that a particular reading is supported by a particular witness. In line 2212a of *Beowulf*, for example, the manuscript is read in different ways by different scholars; the editor Klaeber prints one text, and records in the apparatus two different accounts of the manuscript reading, by Zupitza and Chambers: [1]

```
<l>se &eth;e on
<app>
  <rdg wit='Kl'>hea(um) h(&ae;&th;)e</>
  <rdg wit='MS' resp='Z'>hea&d;o hl&ae;we</>
  <rdg wit='MS' resp='Cha'>heaum hope</>
</app></l>
<l>hord beweotode,</l>
```

The hand and resp attributes are intelligible only on an element recording a reading from a single witness, and should not be used if more than one witness is given on the same <rdg> or <lem> element. If more than one witness is given for the reading, they are undefined. To convey this information when the witness is one among several, the <witDetail> element should be used; see section 19.1.4 ('Witness Information') on p. 475.

[1] For the sake of legibility in the example, long marks over vowels are omitted. The non-standard entities 'ae', 'd' and 'th' refer to a-e ligature, eth, and thorn respectively.

Where there is a greater weight of editorial discussion and interpretation than can conveniently be expressed through the attributes provided on these tags (e.g. multiple causes for a single reading; multiple editorial responsibility for an emendation) this information can be attached to the apparatus in a note, or recorded in the feature structure notation defined in chapter 16 ('Feature Structures') on p. 397. In particular, such recurring text-critical situations as palaeographic confusion of particular letters, or homoeoarchy or homoeoteleuton involving specific character groups, may lend themselves to feature structure treatment. Information concerning these recurrent situations may be encoded into database-like fragments within the text which would then be available to sophisticated computer-assisted analysis. Further work remains to be done on such mechanisms, however, and so no examples are given here of the use of feature structures in text-critical apparatus.

The <note> element may also be used to record the specific wording of notes in the apparatus of the source edition, as here in a transcription of Friedrich Klaeber's note on *Beowulf* 2207a:

```
<l n='2207a'>sy&d;&d;an Beowulfe
  <note place='app' resp='Kl'>Fol. 179a <mentioned>beowulfe</>.
Folio 179, with the last page (Fol. 198b), is the worst
part of the entire MS.  It has been freshened up by a
later hand, but not always correctly.  Information on
doubtful readings is in the notes of Zupitza and Chambers.
</note>
</l>
<l n='2207b'>brade rice</l>
<!-- ... -->
```

Notes providing details of the reading of one particular witness should be encoded using the specialized <witDetail> element described in section 19.1.4 ('Witness Information') on p. 475.

Encoders should be aware of the distinct fields of use of the attribute values wit, hand, and resp. Broadly, wit identifies the physical entity in which the reading is found (manuscript, clay tablet, papyrus, printed edition); hand refers to the agent responsible for inscribing that reading in that physical entity (scribe, author, inscriber, hand 1, hand 2); resp indicates the scholar responsible for asserting the existence of that reading in that physical entity. In some cases, the categories may blur: a scholar may produce an edition introducing readings for which he or she is responsible; that edition may itself become a witness in a later critical apparatus. Thus, readings introduced as corrections in the earlier edition will be seen in the later apparatus as witnessed by the earlier edition. As observed in the discussion concerning the discrimination of hand and resp in transcription of primary sources in section 18.2.2 ('Hand, Responsibility, and Certainty Attributes') on p. 459, the division of layers of responsibility through various scholars for particular aspects of a particular reading may require the more complex mechanisms for assigning responsibility described in chapter 17 ('Certainty and Responsibility') on p. 435.

The formal declaration of the <rdg> and <lem> elements is this:

```
<!-- 19.1.2:  Readings                                          -->
<!ELEMENT lem           - O  (%paraContent)
                                                 +(%m.fragmentary)
                                                              >
<!ATTLIST lem               %a.global;
                            %a.readings;                      >
<!ELEMENT rdg           - O  (%paraContent)
                                                 +(%m.fragmentary)
                                                              >
<!ATTLIST rdg               %a.global;
                            %a.readings;                      >
<!-- This fragment is used in sec. 19                         -->
```

19.1.3 Indicating Subvariation in Apparatus Entries

The <rdgGrp> element may be used to group readings, either because they have identical values on one or more attributes, or because they are seen as forming a self-contained variant sequence, or for some other reason. This grouping of readings is entirely optional: no such grouping of

readings is required.

<rdgGrp> within a textual variation, groups two or more readings perceived to have a genetic relationship or other affinity.

The <rdgGrp> element is a member of class *readings* and therefore can carry the wit, type, cause, varSeg, hand, and resp attributes described in the preceding section. When values for any of these attributes are given on a <rdgGrp> element, the values given are inherited by the <rdg> or <lem> elements nested within the reading group, unless overridden by a new specification on the individual reading element.

To indicate that both Hg and La vary only orthographically from the lemma, one might tag both readings <rdg type='orthographic'>, as shown in the preceding section. This fact can be expressed more perspicuously, however, by grouping their readings into a <rdgGrp>, thus:

```
<app>
    <lem wit='El Ra2'>though</>
    <rdgGrp type=orthographic>
         <rdg wit='Hg'>thogh</>
         <rdg wit='La'>thouhe</>
    </rdgGrp>
</app>
```

Similarly, <rdgGrp> may be used to organize the substantive variants of an apparatus entry. Editors may need to indicate that each of a group of witnesses may be taken as all supporting a particular reading, even though there may be variation concerning the exact form of that reading in, or the degree of support offered by, those witnesses. For example: one may identify three substantive variants on the first word of Chaucer's *Wife of Bath's Prologue* in the manuscripts: these might be expressed in regularized spelling as "Experience", "Experiment", and "Eriment". In fact, the manuscripts display many different spellings of these words, and a scholar may wish both to show that the manuscripts have all these variant spellings and that these variant spellings actually support only the three regularized spelling forms. One may term these variant spellings as "subvariants" of the regularized spelling forms.

This subvariation can be expressed within an <app> element by gathering the readings into three groups according to the normalized form of their reading. All the readings within each group may be accounted subvariants of the main reading for the group, which may be signaled by tagging it <lem> or <rdg type='group base'>.

In this example, the different subvariants on "Experience", "Experiment", and "Eriment" are held within three <rdgGrp> elements nested within the enclosing <app> element:

```
<app type='substantive'>
    <rdgGrp type='subvariants'>
        <lem wit='El Hg'>Experience</>
        <rdg wit='Ha4'>Experiens</>
    </rdgGrp>
    <rdgGrp type='subvariants'>
        <lem wit='Cp Ld1'>Experiment</>
        <rdg wit='La'>Ex&p-underbar;iment</>
    </rdgGrp>
    <rdgGrp type='subvariants'>
        <lem wit='[unattested]'>Eriment</>
        <rdg wit='Ra2'>Eryment</>
    </rdgGrp>
</app>
```

From this, one may deduce that the regularized reading "Experience" is supported by all three manuscripts El Hg Ha4, although the spelling differs in Ha4, and that the regularized reading "Eriment" is supported by Ra2, even though the form differs in that manuscript. Accordingly, an application which recognizes that these apparatus entries show subvariation may then assign all the witnesses instanced as attesting the sub-variants on that lemma as actually supporting the reading of the lemma itself at a higher level of classification. Thus, Ha4 here supports the reading "Experience" found in El and Hg, even though it is spelt slightly differently in Ha4.

Reading groups may nest recursively, so that variants can be classified to any desired depth. Because apparatus entries may also nest, the <app> element might also be used to group readings in the same way. The example above is substantially identical to the following, which uses <app> instead of <rdgGrp>:

```
<app id=A1 type='substantive' >
    <rdg wit='El Hg Ha4'>
        <app id=A2 type='orthographic'>
            <lem wit='El Hg'>Experience</>
            <rdg wit='Ha4'>Experiens</>
        </app>
    </rdg>
    <rdg wit='Cp Ld1 La'>
        <app id=A3 type='orthographic'>
            <lem wit='Cp Ld1'>Experiment</>
            <rdg wit='La'>Ex&p-underbar;iment</>
        </app>
    </rdg>
    <rdg wit='Ra2'>
        <app id=A4 type='orthographic'>
            <lem wit='[unattested]'>Eriment</>
            <rdg wit='Ra2'>Eryment</>
        </app>
    </rdg>
</app>
```

This expresses even more clearly than the previous encoding of this material that at the highest level of classification (apparatus entry A1), this variation has three normalized readings, and that the first of these is supported by manuscripts El, Hg, and Ha4; the second by Cp, Ld1, and La; and the third by Ra2. Some encoders may find the use of nested apparatus entries less intuitive than the use of reading groups, however, so both methods of classifying the readings of a variation are allowed.

Reading groups may also be used to bring together variants which form an apparent developmental sequence, and to make clear that other readings are not part of that sequence, as in the following example, which makes clear that the variant sequence "experiment" to "eriment" says nothing about the relative priority of "experiment" and "experience":

```
<app type='substantive'>
    <rdgGrp type='subvariants'>
        <lem wit='El Hg'>Experience</>
        <rdg wit='Ha4'>Experiens</>
    </rdgGrp>
    <rdgGrp type='sequence'>
        <rdgGrp varSeq=1 type='subvariants'>
            <lem wit='Cp Ld1'>Experiment</>
            <rdg wit='La'>Ex&p-underbar;iment</>
        </rdgGrp>
        <rdgGrp varSeq=2 cause='loss of abbrev for PER' resp=PR>
            <lem wit='[unattested]'>Eriment</>
            <rdg wit='Ra2'>Eryment</>
        </rdgGrp>
    </rdgGrp>
</app>
```

Reading groups are defined formally as follows:

```
<!-- 19.1.3:  Reading Groups                                    -->
<!ELEMENT rdgGrp        - O  (rdgGrp | (rdg, wit?))+             >
<!ATTLIST rdgGrp             %a.global;
                             %a.readings;                        >
<!-- This fragment is used in sec. 19                           -->
```

19.1.4 Witness Information

A given reading is associated with the set of witnesses attesting it by listing the witnesses in the wit attribute on the <rdg>, <lem>, or <rdgGrp> element. Special mechanisms, described in the following sections, are needed to associate annotation on a reading with one specific witness among several (section 19.1.4.1 ('Witness Detail Information') on p. 475), to transcribe witness information verbatim from a source edition (section 19.1.4.2 ('Witness Information in the Source') on p. 476), and to identify the formal lists of witnesses typically provided in the front matter of critical editions (section 19.1.4.3 ('The Witness List') on p. 477).

19.1.4.1 Witness Detail Information

When it is desired to give additional information about a particular witness or witnesses for the reading, the information may be given in a <witDetail> element, pointing to the identifier for that reading and signalling in the value of its wit attribute the witnesses or witnesses to which the additional information relates.

<witDetail> gives further information about a particular witness, or witnesses, to a particular reading. Attributes include:

target indicates the identifier for the reading, or readings, to which the witness detail refers.
wit indicates the sigil or sigla for the witnesses to which the detail refers.

The <witDetail> element is a specialized form of <note>, which adds to the attributes of that element the specialized attribute wit, which indicates which witness in particular is being described. Like <note>, <witDetail> can be included in the text at the point of attachment, or can point to the reading(s) being annotated with its target attribute. To indicate, on the authority of editor PR, that the Ellesmere manuscript has an ornamental capital in the word 'Experience', for example, one might write:

```
<app type='substantive'>
    <rdgGrp type='subvariants'>
        <lem id=W026 wit='El Hg'>Experience</>
        <rdg wit='Ha4'>Experiens</>
    </rdgGrp>
    <!-- ... -->
</app>

<!-- elsewhere in the text, perhaps in a separate section
     of notes ... -->
<witDetail target=w026 resp=PR wit='El'>Ornamental capital.</>
```

This encoding makes clear that the ornamental capital mentioned is in the Ellesmere manuscript, and not in Hengwrt or Ha4.

Like <note>, <witDetail> may be used to record the specific wording of information in the source text, even when the information itself is captured in some more formal way elsewhere. The example from the Carmina Burana above (section 19.1.2 ('Readings') on p. 470), for example, might be extended thus, to record the wording of the note explaining the variant:

```
<l>Swaz hi g&a;t umbe
<l>daz sint alle megede,
<l>die wellent &a;n man
<l><app>
     <rdg wit='M' hand='m1'>alle</>
     <rdg id='Anon.6.4' wit='M' hand='m2'>allen</>
   </app>
   disen sumer g&a;n.

<!-- ... -->
<witDetail target='Anon.6.4' wit='M'><ref>allen</ref>
   <mentioned>n</> nachgetragen.</witDetail>
<!-- ... -->
```

Observe that a single witness detail element may be linked to several different readings (noting, for example, a recurrent phenomena in a particular manuscript) by having the target attribute point at all the readings in question. Similarly, feature structures containing information about the text in a witness (whether retroversion, regularization, or other) can also be linked to specific <lem> and <rdg> instances. See chapter 16 ('Feature Structures') on p. 397.

The <witDetail> element is formally declared thus:

```
<!-- 19.1.4.1:  Witness Details                                  -->
<!ELEMENT witDetail     - O  (%paraContent)                       >
<!ATTLIST witDetail          %a.global;
          target             IDREFS              #REQUIRED
          wit                CDATA               #REQUIRED
          type               CDATA               #IMPLIED
          place              CDATA               'apparatus'    >
<!-- This fragment is used in sec. 19                            -->
```

19.1.4.2 Witness Information in the Source

In the transcription of printed critical editions, it may be desirable to retain for future reference the exact form in which the source edition records the witnesses to a particular reading; this is particularly important in cases of ambiguity in the information, or uncertainty as to the correct interpretation. The <wit> element may be used to transcribe such lists of witnesses to a particular reading.

<wit> contains a list of one or more sigla of witnesses attesting a given reading, in a textual variation.

The <wit> list may appear following a <rdg>, <rdgGrp>, or <lem> element in any apparatus entry, and should be used only to transcribe the witness information in the form found in the source. The advantage of holding witness information in the wit attribute of <lem> or <rdg> is that this may make it more convenient for an SGML application to check that every sigil identifier has been declared elsewhere in the document. By giving the wit attribute a declared value of IDREFS, for example, one could more easily ensure that readings are assigned only to witness sigla given as ID values for witnesses in a <witList> element (see section 19.1.4.3 ('The Witness List') on p. 477). Such checking is somewhat more difficult for witness sigla held as the content of a <wit> element: an application program can check them, but SGML parsers will not. For this reasons, it is recommended that encoders always hold witness information in the wit attribute of <lem> and <rdg>, where possible. Thus, as in the examples below, even when a reference to a witness is exactly reproduced in the <wit> element, the corresponding sigil for that witness can be written into the wit attribute of the matching <rdg> or <lem>. However, in cases where it is uncertain how the witness reference contained in the <wit> element should be interpreted, the wit attribute on the matching <rdg> or <lem> may be left empty.

```
<lg>
<l id=Diet1.1>Sl&a-;fest du, vriedel ziere?
<l id=Diet1.2>wan wecket uns leider schiere;
<l id=Diet1.3>ein vogell&i-;n s&o-; wol get&a-;n
<l id=Diet1.3>daz ist der linden an daz zw&i-; geg&a-;n.
</lg>

<!-- ... -->
<app type='secondary' loc=Diet.1.1>
    <rdg wit='K Ba'>sl&a-;fst</> <wit>K(Ba)</>
    </app>
<app type='secondary' loc=Diet.1.2>
    <rdg wit='K V'>Man</> <wit>K(V)</>
    <rdg wit='K Wa'>weckt</> <wit>K (Wackernagel 401)</>
    <rdg wit='Ju'>Ich waen ez taget uns schiere</>
       <wit><bibl>Jungbluth, Festschr. Pretzel 1963, 122.</></>
    </app>
<!-- ... -->
```

```
<!-- (The non-standard entities &a-; &o-; and &i-; are used   -->
<!-- here to indicate vowels with circumflexes.)              -->
<!-- The edition in question has two apparatus:  one of       -->
<!-- manuscript readings, and one of readings from editions   -->
<!-- and the secondary literature; hence the attribute        -->
<!-- value type='secondary'.                                  -->
```

Of course, the sigla used for different witnesses need not be the same in the source and the wit attribute, as shown particularly in the apparatus for the second line of the poem (Diet.1.2).

The formal declaration for <wit> is as follows:

```
<!-- 19.1.4.2:  Source-text Witness Lists in Apparatus        -->
<!ELEMENT wit           - O  (%phrase.seq;)                    >
<!ATTLIST wit                %a.global;                        >
<!-- This fragment is used in sec. 19                         -->
```

19.1.4.3 The Witness List

In the front matter of the edition, a list of all witnesses may be given if desired, in the form of a witness list, held within a <witList> element. This witness list must contain a series of <witness> elements. Each <witness> element may optionally contain text describing that witness in detail and must have an attibute holding as its value the sigil (siglum) or identifier for a particular witness.

<witList> contains a list of all the witnesses referred to in <wit> elements or wit attributes within the critical apparatus.

<witness> contains either a description of a single witness referred to within the critical apparatus, or a list of witnesses which is to be referred to by a single sigil. Attributes include:

- **sigil** indicates the sigil for one witness or for one group of witnesses to which readings are assigned in a critical apparatus.
- **included** indicates which other witnesses are included in a witness group.

The minimal information provided by a witness list is thus the set of sigla for all the witnesses named in the apparatus. For example, a simple list of the four Chaucer manuscripts used in the examples of this chapter could appear thus:

```
<witlist>
    <witness sigil='El'>
    <witness sigil='Hg'>
    <witness sigil='La'>
    <witness sigil='Ra2'>
</witlist>
```

It is common, however, for witness lists to be somewhat more informative: each <witness> element may contain a prose description of the witness, or a bibliographic citation:

```
<witlist>
    <witness sigil='El' >Ellesmere, Huntingdon Library 26.C.9</witness>
    <witness sigil='Hg' >Hengwrt, National Library of Wales, Aberystwyth,
                Peniarth 392D</witness>
    <witness sigil='La' >British Library Lansdowne 851</witness>
    <witness sigil='Ra2'>Bodleian Library Rawlinson Poetic 149</witness>
</witlist>
```

In some cases, the witness list contains a whole paragraph of commentary for each witness:

```
<witList>
<witness sigil='A'>die sog. <soCalled>Kleine (oder alte)
        Heidelberger Liederhandschrift</>.
        <bibl>Universit&auml;tsbibliothek Heidelberg col. pal.
        germ. 357.  Pergament, 45 Fll. 18,5 &times 13,5 cm.</bibl>
        Wahrscheinlich die &ae;lteste der drei gro&ss;en Hss.
        Sie <q>datiert aus dem 123. Jahrhundert, etwa um 1275.
        Ihre Sprache weist ins Elsa&ss;, evtl. nach Stra&ss;burg.
        Man geht wohl nicht fehl, in ihr eine Sammlung aus dem
```

```
        Stadtpatriziat zu sehen</q> (<bibl><author>Blank</>,
        [vgl. <ref>Lit. z. Hss. Bd. 2, S. 39</ref>] S. 14</bibl>).
        Sie enth&ae;lt 34 namentlich genannte Dichter.  <q>Zu
        den Vorz&ue;gen von A geh&oe;rt, da&ss; sie kaum je
        bewu&ss;t ge&ae;ndert hat, so da&ss; sie f&ue;r manche
        Dichter ... oft den besten Text liefert</q> (so wohl
        mit Recht <bibl><author>v. Kraus</></>).</witness>
<witness sigil='a'>Bezeichnung <bibl><author>Lachmann</></>s f&ue;r
        die von einer 2. Hand auf bl. 40-43 geschriebenen
        Strophen der Hs. A.</witness>
<witness sigil='B'>die <soCalled>Weingartner (Stuttgarter)
        Liederhandschrift</>.  <bibl>W&uerttembergische
        Landesbibliothek Stuttgart, HB XIII poetae germanici
        1.  Pergament, 156 Bll.  15 &times 11,5 cm; 25 teils
        ganzseitig, teils halbseitige Miniaturen.</bibl>  Kaum
        vor 1306 in Konstanz geschrieben.  Sie enth&ae;lt Lieder
        von 25 namentlich genannten Dichtern.  (Dazu kommen
        Gedichte von einigen ungenannten bzw. unbekannten
        Dichtern, ein Marienlobpreis und eine Minnelehre.)</witness>
<!-- etc. ... -->
</witList>

<!-- &ae; &oe; &ue; = a, o, u umlaut. ss = esszett            -->
```

It is common, in text-critical work, to refer to frequently occurring groups of witnesses by means of a single common sigil. Such sigla may be documented as pseudo-witnesses in their own right by including, in the witness list, a <witness> element giving the sigil for the group and listing the other witnesses included in the group in the value of the included attribute. In this example, the group of manuscripts of the "Canterbury Tales" which make up "Constant Group c" are themselves first allocated sigla in individual <witness> elements, and then those sigla are given as the included value of a further <witness> element. All the manuscripts of this group may thereafter be referred to as "c":

```
<witlist>
    <witness sigil='Cp'>Corpus Christi Oxford MS 198</witness>
    <witness sigil='La'>British Library Lansdowne 851</witness>
    <witness sigil='Sl2'>British Library Sloane MS 1686</witness>

    <witness sigil='c' included='Cp La Sl2'>Constant Group c</>
</witlist>
```

That the reading "Experiment" occurs in all three manuscripts can now be indicated simply as follows:

```
<rdg wit='c'>Experiment</rdg>
```

Situations commonly arise where there are many more or less fragmentary witnesses, such that there may be quite distinct groups of witnesses for different parts of a text or collection of texts. One may treat this with distinct <witList> elements for each different part. Alternatively, one may have a single <witList> element at the beginning of the file listing all the witnesses, partial and complete, for the text, with the attestation of fragmentary witnesses indicated within the apparatus by use of the <witStart> and <witEnd> elements described in section 19.1.5 ('Fragmentary Witnesses') on p. 479.

If a witness list is provided, it may be unnecessary to give, in each apparatus entry, an exhaustive list of the witnesses which agree with the base text. An application program can — in principle — compare the witnesses given for each variant found with those given in the full list of witnesses, subtracting from this list all the witnesses not active at this point (perhaps because of lacuna, or because they contain a variation on a different, overlapping lemma) and thence calculate all the manuscripts agreeing with the base text. In practice, encoders may find it less error-prone to list all witnesses explicitly in each apparatus entry.

The formal declaration of <witList> and <witness> is as follows:

```
<!-- 19.1.4.3:  Witness Lists in Front Matter                -->
<!ELEMENT witList       - O  (witness+)                       >
<!ATTLIST witList            %a.global;                       >
<!ELEMENT witness       - O  (%paraContent)                   >
<!ATTLIST witness            %a.global;
          sigil              CDATA              #REQUIRED
          included           CDATA              ''            >
<!-- This fragment is used in sec. 19                         -->
```

19.1.5 Fragmentary Witnesses

If a witness is incomplete (whether a single fragment, a series of fragments, or a relatively complete text with one or more lacunae), it is usually desirable to record explicitly where its preserved portions begin and end. The following empty tags, which may occur within any <lem> or <rdg> element, indicate the beginning or end of a fragmentary witness or of a lacuna within a witness:

<witStart> indicates the beginning, or resumption, of the text of a fragmentary witness.

<witEnd> indicates the end, or suspension, of the text of a fragmentary witness.

<lacunaStart> indicates the beginning of a lacuna in the text of a mostly complete textual witness.

<lacunaEnd> indicates the end of a lacuna in a mostly complete textual witness.

All are members of the model class *fragmentary*.

Suppose a fragment of a manuscript X of the *Wife of Bath's Prologue* has a physical lacuna, and the text of the manuscript begins with 'auctorite'. In an apparatus this might appear thus, distinguished from the reading of other manuscripts by the presence of the <lacunaEnd> element:

```
<app>
    <lem wit='El Hg'>Auctoritee</lem>
    <rdg wit='La Ra2'>auctorite</rdg>
    <rdg wit='X'><lacunaEnd>auctorite</rdg>
</app>
```

In some cases, the apparatus in the source may commence recording the readings for a particular witness without its being clear whether the previous absence of readings for this witness is due to a lacuna, or to some other reason. The <witStart> element may be used in this circumstance:

```
<app>
    <lem wit='El Hg'>Auctoritee</>
    <rdg wit='La Ra2'>auctorite</rdg>
    <rdg wit='X'><witStart>auctorite</>
</app>
```

The formal declarations for these elements are these:

```
<!-- 19.1.5:  Fragmentary witnesses                           -->
<!ELEMENT witStart      - O  EMPTY                            >
<!ATTLIST witStart           %a.global;
                             %a.fragmentary;                  >
<!ELEMENT witEnd        - O  EMPTY                            >
<!ATTLIST witEnd             %a.global;
                             %a.fragmentary;                  >
<!ELEMENT lacunaStart   - O  EMPTY                            >
<!ATTLIST lacunaStart        %a.global;
                             %a.fragmentary;                  >
<!ELEMENT lacunaEnd     - O  EMPTY                            >
<!ATTLIST lacunaEnd          %a.global;
                             %a.fragmentary;                  >
<!-- This fragment is used in sec. 19                         -->
```

19.2 Linking the Apparatus to the Text

Three different methods may be used to link a critical apparatus to the text:

- the location-referenced method,
- the double-end-point-attached method, and
- the parallel segmentation method.

Both the location-referenced and the double end-point methods may be used with either *in-line* or *external* apparatus, the former dispersed within the base text, the latter held in some separate location, within or outside the document with the base text. The parallel segmentation method does not use the concept of a base text and may only be used for in-line apparatus.

Any document containing <app> elements requires a <variantEncoding> declaration in the <editorialDecl> element of its TEI header, thus:

<variantEncoding> declares the method used to encode text-critical variants. Attributes include:

method indicates which method is used to encode the apparatus of variants. Legal values are:

location-referenced apparatus uses line numbers or other canonical reference scheme referenced in a base text.

double-end-point apparatus indicates the precise locations of the beginning and ending of each lemma relative to a base text.

parallel-segmentation alternate readings of a passage are given in parallel in the text; no notion of a base text is necessary.

location indicates whether the apparatus appears within the running text or external to it. Legal values are:

internal apparatus appears within the running text.

external apparatus appears outside the base text.

For examples of this element, see the following sections. The formal declaration is given in section 5.3.3 ('The Editorial Practices Declaration') on p. 96.

19.2.1 The Location-referenced Method

The location-referenced method of encoding apparatus provides a convenient method for encoding printed apparatus; in this method as in most printed editions, the apparatus is linked to the base text by indicating explicitly only the block of text on which there is a variant (noted usually by a canonical reference scheme, or by line number in the edition, such as "A 137" or "Page 15 line 1").

If the location-referenced method is used for an apparatus stored externally to the base text, the TEI header must have the declaration:

```
<variantEncoding method='location-referenced' location='external'>
```

In the <body> of the document, the base text (here El) will appear:

```
<text>
<body> ...
<div n=WBP><head>The Prologe of the Wyves Tale of Bathe</head>
<l n=1>Experience though noon Auctoritee</l>
<l>Were in this world ...
</div>
</body>
</text>
```

Elsewhere in the document, or in a separate file, the apparatus will appear. On each <app> element, the loc attribute should be specified to indicate where the variant occurs in the base text.

```
<app loc="WBP 1">
   <rdg wit='La'>Experiment</>
   <rdg wit='Ra2'>Eryment</>
</app>
```

If the same text is encoded using in-line storage, the apparatus is dispersed through the base text block to which it refers. In this case, the location of the variant can be read from the line in which it occurs.

```
<TEI.2>
<TeiHeader>
  <!-- ... -->
  <encodingDesc>
    <!-- ... -->
    <editorialDecl>
      <!-- ... -->
      <variantEncoding method='location-referenced' location='internal'>
      <!-- ... -->
    </editorialDecl>
    <!-- ... -->
  </encodingDesc><!-- ... -->
</TeiHeader>
<text>
<body>
  <!-- ... -->
  <div n=WBP><head>The Prologe ... </head>
     <l n=1>Experience
     <app>
          <rdg wit='La'>Experiment</>
          <rdg wit='Ra2'>Eryment</>
     </app>
     though noon Auctoritee</l>
     <l>Were in this world ...
</body>
</text>
</TEI.2>
```

Since the location is not required to be exact, the apparatus for a line might also appear at the end of the line:

```
<l n=1>Experience though noon Auctoritee
<app>
    <rdg wit='La'> Experiment</>
    <rdg wit='Ra2'> Eryment</>
</app></l>
<l>Were in this world ...
```

When the apparatus is linked to the text by means of location references, as shown here, it is not possible to find automatically the precise portion of text varied by the readings. In order to show explicitly what portion of the base text is replaced by the variant readings, the <lem> element may be used:

```
<l n=1>Experience though noon Auctoritee
<app>
    <lem wit='El' >Experience</>
    <rdg wit='La' >Experiment</>
    <rdg wit='Ra2'>Eryment</>
</app></l>
<l>Were in this world ...
```

Often the lemma will have no attributes, being simply the "base-text reading" and requiring no qualification, but it may optionally carry the normal attributes, as shown here. Some text critics prefer to abbreviate or elide the lemma, in order to save space or trouble; such practice is not forbidden by these Guidelines, but no recommendations are made for conventions of

abbreviating the lemma, whether abbreviation of each word, or suppression of all but the first and last word, etc.

Where it is intended that the apparatus be complete enough to allow the "reconstruction" of the witnesses (or at least of their non-orthographic variations), the location-reference method should be avoided in favor of one of the other two methods, which allow the unambiguous reconstruction of the lemma from the encoding.

19.2.2 The Double End-Point Attachment Method

In the double end-point attachment method, the beginning and end of the lemma in the base text are both explicitly indicated. It thus differs from the location-referenced method, in which only the larger span of text containing the lemma is indicated. Double end-point attachment permits unambiguous matching of each variant reading against its lemma. It or the parallel-segmentation method should be used in all cases where this is desired, for example where the apparatus is intended to enable full reconstruction of the text, or of the substantives, of every witness.

When the double endpoint attachment method is used, the from and to attributes of the <app> element are used to indicate the beginning and ending points of the reading in the base text: their values are SGML identifiers which occur at the locations in question. If no other markup is present there, the beginning and ending points should be marked using the <anchor> element defined in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331. In cases where it is not possible to insert anchors within the base text (e.g. where the text is on a read-only medium) the beginning and end of the lemma may be indicated by using the "indirect pointing" mechanisms discussed in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331. Explicit anchors are more likely to be reliable, and are therefore to be preferred.

The double end-point attachment method may be used with in-line or external apparatus. In the latter case, the base text (here El) will appear with <anchor> elements inserted at every place where a variant begins or ends (unless some element with an SGML identifier already begins or ends at that point):

```
<TEI.2>
<teiHeader>
    <!-- ... -->
    <variantEncoding method='double-end-point' location='external'>
    <!-- ... -->
</TeiHeader>
<text>
<body>
 <!-- ... -->
 <div n=WBP><head>The Prologe ... </head>
 <l n=1 id='WBP.1'>Experience <anchor id=a2> though noon
        Auctoritee</l>
 <l>Were in this world ...
  <!-- ... -->
</text>
</TEI.2>
```

The apparatus will be separately encoded:

```
<app from=wbp.1 to=a2>
    <rdg wit='La'>Experiment</>
    <rdg wit='Ra2'>Eryment</>
</app>
```

No <anchor> element is needed at the beginning of the line, since the from attribute can use the SGML identifier for the line as a whole; the lemma is assumed to run from the beginning of the element indicated by the from attribute, to the end of that indicated by the to attribute. If no value is given for to, the lemma runs from the beginning to the end of the element indicated by the from attribute.

When the apparatus is encoded in-line, it is dispersed through the base text. Only the

beginning of the lemma need be marked with an <anchor>, since the <app> is inserted at the end of the lemma, and itself therefore marks the end of the lemma.

```
<TEI.2>
<teiHeader>
    <!-- ... -->
    <variantEncoding method='double-end-point' location='internal'>
    <!-- ... -->
</TEI.Header>
<text>
<body>
<!-- ... -->
<div><head>The Prologe ...</head>
<l n=1 id=WBP.1>Experience
     <app from=WBP.1>
          <rdg wit='La'>Experiment</>
          <rdg wit='Ra2'>Eryment</>
     </app>
though noon Auctoritee
<l>Were in this world ...
<!-- ... -->
</body>
</TEI.2>
```

The lemma need not be repeated within the <app> element in this method, as it may be extracted reliably from the base text. If an exhaustive list of witnesses is available, it will also not be necessary to specify just which manuscripts agree with the base-text to enable reconstruction of witnesses. An application will be able to determine the manuscripts that witness the base reading, by noting which witnesses are attested as having a variant reading, and inferring the base-text reading for all others after adjusting for fragmentary witnesses and for witnesses carrying overlapping variant readings.

Alternatively, if it is desired to make an explicit record of the attestation of the base text the <lem> element may be embedded within <app>, carrying the witnesses to the base. Thus

```
<app  from=wbp.1 to=a2>
    <lem wit='El Hg'>Experience</>
    <rdg wit='La'>Experiment</>
    <rdg wit='Ra2'>Eryment</>
</app>
```

This method is designed to cope with "overlapping lemmata". For example, at line 117 of the Wife of Bath's Prologue, the manuscripts Hg (Hengwrt), El (Ellesmere) and Ha4 (British Library Harleian 7334) read:

Hg: And of so parfit wys a wight ywroght
El: And for what profit was a wight ywroght
Ha4: And in what wise was a wight ywroght

In this case, one might wish to record "in what wise was" in Ha4 as a single variant for "of so parfit wys" in Hg, and "was a wight" in El and H4 as a variant on "wys a wight" in Hg. This method can readily cope with such difficult situations, typically found in large and complex traditions:

```
<l n=117 id='WBP.117' > And
   <anchor id='a117.1'> of so parfit
   <anchor id='a117.2'> wys
   <anchor id='a117.3'> a wight
   <anchor id='a117.4'> ywroght</l>

<!-- ... -->
<app from=a117.1 to=a117.3>
     <lem wit='Hg' >of so parfit wys</>
     <rdg wit='Ha4'>in what wise was</>
</app>
```

```
<app from=a117.2 to=a117.4>
     <lem wit='Hg'>wys a wight</>
     <rdg wit='El Ha4'>was a wight</>
</app>
```

The parallel segmentation method, to be discussed next, cannot handle overlaps among variants, and would require the individual variants to be split into pieces.

Because creation and interpretation of double end-point attachment apparatus will be lengthy and difficult it is likely that they will usually be created and examined by scholars only with mechanical assistance.

19.2.3 The Parallel Segmentation Method

This method differs from the double end-point attachment method in that all variants at any point of the text are expressed as variants on one another. In this method, no two variations can overlap, although they may nest. Thus, the concepts of a base text and of a lemma become unnecessary: the texts compared are divided into matching segments all synchronized with one another. This permits direct comparison of any span of text in any witness with that in any other witness. It is also very easy with this method for an application to extract the full text of any one witness from the apparatus.

This method will (by definition) always be satisfactory when there are just two texts for comparison (assuming they are in the same language and script). It will also be useful where editors do not wish to privilege a text as the "base" or when editors wish to present parallel texts. It will become less convenient as traditions become more complex and tension develops between the need to segment on the largest variation found and the need to express the finest detail of agreement between witnesses.

In the parallel segmentation method, each segment of text on which there is variation is marked by an <app> element; each reading is given in a <rdg> element; if it is desired to single out one reading as preferred, it may be tagged <lem>:

```
<TEI.2><teiHeader>
<!-- ... -->
    <variantEncoding method='parallel-segmentation'
                     location='internal'>
    <!-- ... -->
</teiHeader>
<text>
<body>
<!-- ... -->
    <l n=1>
    <app><lem wit='El Hg'>Experience</>
         <rdg wit='La'>Experiment</>
         <rdg wit='Ra2'>Eryment</>
    </app>
    though noon Auctoritee</l>
    <l>Were in this world ...
```

This method cannot be used with external apparatus: it must be used in-line. Note that apparatus encoded with this method may be translated into the double end-point attachment method and back without loss of information. Where double-end-point-attachment encodings have no overlapping lemmata, translation of these to the parallel segmentation encoding and back will also be possible without loss of information.

For economy, the witnesses to the reading most widely attested need not be stated. Since all manuscripts must be represented in all apparatus entries, it will be possible for an application to read a <witList> declaring all the witnesses to the text and then calculate which witnesses have not been named. In the example below, only La and Ra2 are identified explicitly with a reading; an application might successfully infer from this that 'Experience', whose witnesses are not given, must be attested by El and Hg. To avoid confusion, however, witnesses may be omitted only for a single reading.

```
<l n=1>
<app><lem>Experience</>
     <rdg wit='La'>Experiment</>
     <rdg wit='Ra2'>Eryment</>
</app>
though noon Auctoritee</l>
<l>Were in this world ...
```

Alternatively, the witnesses for every reading may be stated, as in the first example.

As noted, apparatus entries may nest in this method: if an imaginary fifth manuscript of the text read "Auctoritee, though none experience", the variation on the individual words of the line would nest within that for the line as a whole:

```
<l n=1>
<app>
  <rdg wit=Chi3>Auctoritee, though none experience</rdg>
  <rdg>
    <app><rdg wit='El Hg'>Experience</>
         <rdg wit='La'>Experiment</>
         <rdg wit='Ra2'>Eryment</>
    </app>
    <app><rdg wit='El Ra2'>though</>
         <rdg wit='Hg'>thogh</>
         <rdg wit='La'>thouh</>
    </app>
    <app><rdg wit='El Hg'>noon Auctorite</>
         <rdg wit='La Ra2'>none auctorite</>
    </app>
  </rdg>
  </app>
</l>
```

Parallel segmentation cannot, however, deal very gracefully with variants which overlap without nesting: such variants must be broken up into pieces in order to keep all witnesses synchronized.

19.3 Using Apparatus Elements in Transcriptions

It is often desirable to record different transcriptions of the one stretch of text. These variant transcriptions may be grouped within a single <app> element. An application may then construct different "views" of the transcription by extraction of the appropriate variant readings from the apparatus elements embedded in the transcription.

For example, alternative expansions can be recorded in several different <expan> elements, all grouped within an <app> element. Consider, for example, the three different transcriptions given below of line 105 of the Hengwrt manuscript of Chaucer's *The Wife of Bath's Prologue*. The last word of the line "Virginite is grete perfection" is written "perfectio" followed by two minims over which a bar has been drawn, which has been read in different ways by different scholars. The first transcription, by Elizabeth Solopova, represents the two minims with bar above by reference to an entity *i-i*. This transcription notes this as a mark of abbreviation but gives no expansion for it. A second transcriber, F. J. Furnivall, regards the bar as an abbreviation of 'u', reading the two minims as an 'n'. A third transcriber, P. G. Ruggiers, regards the bar as an abbreviation of 'n', reading the minims as 'u'. This information may be held within an <app> structure, as follows:

```
Virginite is grete
<app>
   <rdg resp=ES >perfectio<abbr>&i-i;</abbr></rdg>
   <rdg resp=FJF>perfectio<expan>u</>n</rdg>
   <rdg resp=PGR>perfectiou<expan>n</></rdg>
</app>
```

This example illustrates the adaptation of the <rdg> element for use within the transcription of a particular witness. The wit attribute, which may be compulsory in recording variant readings of many witnesses within a critical apparatus, is redundant when recording variant readings relating to a single witness. However, it may be desirable to specify the editorial responsibility for a particular reading within a transcription. For all three readings, the resp attribute on <rdg> assigns this responsibility. Using this system, it will be straightforward for an application to extract from the one file the three different transcriptions done by these scholars. To do this, the application has to look only at the resp attribute on each <rdg> element.

Observe too that in this example the resp attribute is attached to the outer <rdg> element and is not repeated for the inner <expan> elements. There is no need for repetition of the resp attribute values, as the <expan> elements contained within each <rdg> element will inherit the value of the resp from the outer <rdg> element. Thus, the processor will know that the responsibility for the expansion "perfectio<expan>u</>n" lies with "FJF", as "FJF" was responsible for the reading containing this expansion. This simplifies the processing of the information, as the application has only to look at the attribute values for each reading in turn and not for those for elements nested within.

Editorial notes may also be attached to <app> structures within transcriptions. Here, editorial preference for Ruggiers' expansion and an explanation of that preference is given:

```
Virginite is grete
<app>
   <rdg resp=ES>perfecti<abbr>o&i-i;</abbr></rdg>
   <rdg id=F105 resp=FJF>perfectio<expan>u</>n</rdg>
   <rdg id=R105 resp=PGR>perfectiou<expan>n</></rdg>
</app>

<!-- ... note element appearing elsewhere in the document ... -->

<note target="R105 F105">Furnivall's expansion implies
that the bar is an abbreviation for 'u'.  There are no
certain instances of this mark as an abbreviation for 'u'
in these MSS and it is widely used as an abbreviation
for 'n'.  Ruggiers' expansion is to be accepted.</>
```

In most cases, elements used to indicate features of a primary textual source may be represented within an <app> structure simply by nesting them within its readings, just as the <abbr> and <expan> elements are nested within the <rdg> elements in the example just given. However, in cases where the tagged feature extends across a span of text which might itself contain variant readings which it is desired to represent by <app> structures, some adaptation of the tagging may be necessary. For example, a span of text may be marked in the transcription of the primary source as a single deletion but it may be desirable to represent just a few words from this source as individual deletions within the context of a critical apparatus drawing together readings from this and several other witnesses. In this case, the tagging of the span of words as one deletion may need to be decomposed into a series of one-word deletions for encoding within the apparatus. If it is important to record the fact that all were deleted by the same act, the markup may use the <join> element or the next and prev attributes defined by chapter 14 ('Linking, Segmentation, and Alignment') on p. 331.

Chapter 20

Names and Dates

This chapter describes an additional tag set which may be used for the encoding of proper names and other phrases descriptive of persons, places, organizations, and also of dates and times, in a manner more detailed than that possible using the elements already provided for these purposes in the core tag set described in chapter 6 ('Elements Available in All TEI Documents') on p. 119.

In section 6.4 ('Names, Numbers, Dates, Abbreviations, and Addresses') on p. 132 it was noted that the elements provided in the core allow the encoder to specify that a given text segment is a proper noun, or a *referring string*, and to specify the kind of object named or referred to only by supplying a value for the `type` attribute. The elements provided by the present tag set allow the encoder both to supply a detailed sub-structure for such referring strings, and also to distinguish explicitly between names of persons, places or organizations.

Similarly, the elements provided here allow the encoder to supply a detailed analysis of the component parts of any expression which denotes a date or time, which is not possible using the elements described in section 6.4.4 ('Dates and times') on p. 137.

It should be noted however that no provision is made by the present tag set for the representation of the abstract structures, or " virtual objects" to which names or dates may be said to refer. In simple terms, where the core tag set allows one to represent a *name*, this additional tag set allows one to represent a *personal name*, but neither provides for the direct representation of a *person*. Appropriate mechanisms for the encoding of such interpretative gestures may be found in chapters 15 ('Simple Analytic Mechanisms') on p. 381 and 16 ('Feature Structures') on p. 397.

To enable the additional tag set described in the present chapter, a parameter entity *TEI.names.dates* must be declared in the document type subset with the value `INCLUDE`, as further described in section 3.3 ('Invocation of the TEI DTD') on p. 39. A document using the prose base tag set and this additional tag set will thus begin as follows:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
     <!ENTITY % TEI.prose 'INCLUDE' >
     <!ENTITY % TEI.names.dates 'INCLUDE' >
]>
```

The chapter begins by discussing additional tags for the encoding of component parts of personal names (section 20.1 ('Personal Names') on p. 488, place names (section 20.2 ('Place Names') on p. 493) and organizational names (section 20.3 ('Organization Names') on p. 497). Detailed encoding of dates and times is described in section 20.4 ('Dates and Time') on p. 499.

The additional tag set for names and dates, included in the file *teind2.dtd*, has the following overall structure:

```
<!-- 20:  Additional tags for names and dates                -->
<!-- ... declarations from section 20.1                      -->
<!--     (Personal names)                                    -->
<!--     go here ...                                         -->
<!-- ... declarations from section 20.2.3                    -->
<!--     (Names for places)                                  -->
<!--     go here ...                                         -->
<!-- ... declarations from section 20.3                      -->
```

```
<!--     (Organization names)                                    -->
<!--     go here ...                                             -->
<!-- ... declarations from section 20.4.2                       -->
<!--     (Date components)                                       -->
<!--     go here ...                                             -->
<!-- <dtdref dtdfrag=dndmeas> -->
```

When this tag set is enabled, three additional element classes called *persPart*, *placePart*, and *temporalExpr* are declared. The parameter entities corresponding with these classes are declared in the file *teind2.ent*, as follows:

```
<!-- 20:  Additional classes for names and dates             -->
<!ENTITY % x.personPart ''                                     >
<!ENTITY % m.personPart '%x.personPart addName | forename |
          genName | nameLink | roleName | surname'            >
<!ENTITY % x.placePart ''                                      >
<!ENTITY % m.placePart '%x.placePart bloc | country | distance
          | geog | geogName | offset | placeName | region |
          settlement'                                          >
<!ENTITY % x.temporalExpr ''                                   >
<!ENTITY % m.temporalExpr '%x.temporalExpr dateStruct | day |
          distance | hour | minute | month | occasion | offset
          | second | timeStruct | week | year'                >
<!ENTITY % a.names '
          key               CDATA              #IMPLIED
          reg               CDATA              #IMPLIED'     >
<!ENTITY % a.personPart '   %a.names;
          type              CDATA              #IMPLIED
          full              (yes | abb | init) yes
          sort              NUMBER             #IMPLIED'     >
<!ENTITY % a.placePart '    %a.names;
          type              CDATA              #IMPLIED
          full              (yes | abb | init) yes'          >
<!ENTITY % a.temporalExpr '
          value             CDATA              #IMPLIED
          type              CDATA              #IMPLIED
          reg               CDATA              #IMPLIED
          full              (yes | abb | init) yes'          >
```

20.1 Personal Names

The core <rs> and <name> elements can distinguish names in a text but are insufficiently powerful to mark their internal components or structure. To conduct nominal record linkage or even to create an alphabetically sorted list of personal names, it is important to distinguish between a family name, a forename and an honorary title. Similarly, when confronted with a referencing string such as "John, by the grace of God, king of England, lord of Ireland, duke of Normandy and Aquitaine, and count of Anjou", the analyst will often wish to distinguish amongst components giving some hint as to the status, occupation or residence of the person to whom the name belongs. The following elements are provided for these and related purposes:

<persName> contains a proper noun or proper-noun phrase referring to a person, possibly including any or all of the person's forename, surname, honorofic, added names, etc.

<surname> contains a family (inherited) name, as opposed to a given, baptismal, or nick name.

<forename> contains a forename, given or baptismal name.

<roleName> contains a name component which indicates that the referent has a particular role or position in society, such as an official title or rank.

<addName> contains an additional name component, such as a nickname,

epithet, or alias, or any other descriptive phrase used within a personal name.

<nameLink> contains a connecting phrase or link used within a name but not regarded as part of it, such as "van der" or "of".

<genName> contains a name component used to indicating generational information, such as "Junior", or a number used in a monarch's name.

As members of the *names* class, all of these elements share the following attributes:

key provides an alternative identifier for the object being named, such as a database record key.

reg gives a normalized or regularized form of the name used.

Additionally, all of the above elements except for <persName> are members of the class *personPart*, and thus share the following attributes:

type provides more culture- linguistic- or application- specific information used to categorize this name component.

full indicates whether the name component is given in full, as an abbreviation or simply as an initial. Legal values are:

yes the name component is spelled out in full.

abb the name component is given in an abbreviated form.

init the name component is indicated only by one initial.

sort specifies the sort order of the name component in relation to others within the personal name.

The <persName> element may be used in preference to the general <name> element irrespective of whether or not the components of the personal name are also to be marked. Its key and reg attributes are used in exactly the same way as those on the <rs> and <name> elements (see section 6.4 ('Names, Numbers, Dates, Abbreviations, and Addresses') on p. 132). The tag <persName> is synonymous with the tag <name type=person>, except that its type attribute allows for further subcategorization of the personal name for example as a "married", "maiden", "pen", "pseudo" or "religious" name. Consequently the following examples are equivalent:

```
That silly man <rs key=DPB1 type=person reg='Brown, David
Paul'>David Paul Brown</rs>, has suffered the furniture of
his office to be seized the third time for rent.

That silly man
<rs key=DPB1 type=person reg='Brown, David Paul'>
  <name>David Paul Brown</name>
</rs> has suffered the furniture of
his office to be seized the third time for rent.

That silly man
<name key=DPB1 type=person reg='Brown, David Paul'>
David Paul Brown</name> has suffered ...

That silly man
<persName key=DPB1 type=person reg='Brown, David Paul'>
  David Paul Brown</persName> has suffered ...
```

The <persName> element is more powerful than the <rs> and <name> elements because distinctive name components occurring within it can be marked as such.

Many cultures distinguish between a family or inherited *surname* and additional personal names, often known as *given names*. These should be tagged using the <surname> and <forename> elements respectively and may occur in any order:

```
<persName key=FDR1>
  <surname>Roosevelt</surname>,
  <forename>Franklin</forename>
  <forename>Delano</forename>
```

```
</persName>

<persName key=FDR1>
  <forename>Franklin</forename>
  <forename>Delano</forename>
  <surname>Roosevelt</surname>
</persName>
```

The type attribute may be used with both <forename> and <surname> elements to provide further culture- or project- specific detail about the name component, for example:

```
<persName key=FDR1>
 <forename type='first'>Franklin</forename>
 <forename type='middle'>Delano</forename>
 <surname type='last'>Roosevelt</surname>
</persName>

<persName key=MRT1>
 <forename type='Christian'>Margaret</forename>
 <forename type='unused'>Hilda</forename>
 <surname type='maiden'>Roberts</forename>
 <surname type='married'>Thatcher</surname>
</persName>

<persName key=MUAL1 type=religious>
 <forename>Muhammad</forename>
 <surname>Ali</surname>
</persName>
```

In the following two examples the type attribute of the <surname> element is used to indicate so-called double-barrelled or hyphenated surnames:

```
<persName key=KHS1>
  <forename>Kara</forename>
  <surname type=combine>Hattersley-Smith</surname>
</persName>

<persName key=NSJS1>
   <forename>Norman</forename>
   <surname type=combine>St John Stevas</surname>
</persName>
```

In most cases, patronymics should be treated as forenames, thus:

```
 ... but it remained for
 <persName>
   <forename>Snorri</>
   <forename>Sturluson</>
 </persName>
 to combine the two traditions in cyclic form.
```

When a patronymic is used as a surname, however (e.g. by an individual who otherwise would have no surname, but lives in a culture which requires surnames), it may be tagged as such:

```
Even <persName><forename>Finnur</> <surname>Jonsson</>
</persName> acknowledged the artificiality of the
procedure:  <q>As <title>Nj&aacute;la</> now begins,
no original saga ever began.</q>
```

In the following example, the type attribute is used to distinguish a patronymic from other forenames:

```
<persName key=pn9>
  <forename sort=2>Sergei</forename>
  <forename sort=3 type='patronym'>Mikhailovic</forename>
```

```
  <surname sort=1>Uspensky</surname>
</persName>
```

This example also demonstrates the use of the sort attribute common to all members of the *personPart* class; its effect is to state the sequence in which <forename> and <surname> elements should be combined when constructing a sort key for the name.

Some names include generational or dynastic information, such as "Junior" or "senior", or a number: the <genName> element may be used to distinguish these from other parts of the name, as in the following examples:

```
<persName key=HEMA1>
  <surname>Marques</surname>
  <genName>Junior</genName>,
  <forename>Henrique</forename>
</persName>

<persName>
  <foreName>Charles</foreName>
  <genName>II</genName>
</persName>
```

It is also often convenient to distinguish phrases (historically similar to the generational labels mentioned above) used to link parts of a name together, such as "von", "of", "de" etc. It is often a matter of arbitrary choice whether or not such components are regarded as part of the surname or not; the <nameLink> element is provided as a means of making clear what the correct usage should be in a given case, as in the following examples:

```
<persName key=DUDO1>
 <roleName type=honorific full=abb>Mme</roleName>
 <nameLink>de la</nameLink>
 <surname>Rochefoucault</surname>
</persName>

<persName>
  <foreName>Walter</foreName>
   <surname>de la Mare</surname>
</persname>
```

Finally, the <addName> and <roleName> elements are used to mark all name components other than those already listed. The distinction between them is that a <roleName> encloses an associated name component such as an aristocratic or official title which exists in some sense independently of its bearer. The distinction is not always a clear one. As elsewhere, the type attribute may be used with either element to supply culture- or application- specific distinctions. Some typical values for this attribute for names in the Western European tradition follow:

nobility An inherited or life-time title of nobility such as Lord, Viscount, Baron, etc.
honorific An academic or other honorific prefixed to a name e.g. Dr., Professor, Mrs., etc.
office Membership of some elected or appointed organization such as President, Governor, etc.
military Military rank such as Colonel.
epithet A traditional descriptive phrase or nick-name such as The Hammer, The Great, etc.

Here are some further examples of the usage of these elements:

```
<persName key=PGK1>
<roleName type=nobility>Princess</roleName>
<forename>Grace</forename>
</persName>

<persName key=GRMO1 type=pseudo>
<addName type=honorific>Grandma</addName>
<surname>Moses</surname>
</persName>
```

```
<persName key=MRSRO1>
<addName type=honorific>Mrs</addName>
<surname>Robinson</surname>
</persName>

<persName key=STAU1>
<roleName type=office>Saint</addName>
<forename>Augustine</forename>
</persName>

<persName key=SLWICL1>
<roleName type=office>President</roleName>
<foreName>Bill</foreName>
<surname>Clinton</surname>
</persName>

<persName key=MOGA1>
<roleName type=military>Colonel</roleName>
<surname>Gaddafi</surname>
</persName>

<persName key=FRTG1>
<forename>Frederick</forename>
<addname type=epithet>the Great</addname>
</persName>
```

A name may have any combination of the above elements:

```
<persName key=EGBR1>
<roleName type=office>Governor</roleName>
<forename sort=2>Edmund
<forename sort=3 full=init reg='Gerald'>G</forename>.
<addName type=epithet>Jerry</addName>
<addName type=epithet>Moonbeam</addName>
<surname sort=1>Brown</surname>
<relationship full=abb>Jr</relationship>.
</persName>
```

Although highly flexible, these mechanisms for marking personal name components will not cater for every personal name and processing need. Where the internal structure of personal names is highly complex or where name components are particularly ambiguous, feature structures are recommended as the most appropriate mechanism to mark and analyze them, as further discussed in chapter 16 ('Feature Structures') on p. 397.

The elements discussed in this section are formally defined as follows:

```
<!-- 20.1:  Personal names                                    -->
<!ELEMENT persName      - -  (%m.personPart; | %m.phrase; |
                              #PCDATA)*                        >
<!ATTLIST persName           %a.global;
                             %a.names;
          type               CDATA              #IMPLIED      >
<!ELEMENT surname       - -  (%phrase.seq;)                   >
<!ATTLIST surname            %a.global;
                             %a.personPart;                   >
<!ELEMENT forename      - -  (%phrase.seq;)                   >
<!ATTLIST forename           %a.global;
                             %a.personPart;                   >
<!ELEMENT genName       - -  (%phrase.seq;)                   >
<!ATTLIST genName            %a.global;
                             %a.personPart;                   >
<!ELEMENT nameLink      - -  (%phrase.seq;)                   >
<!ATTLIST nameLink           %a.global;
                             %a.personPart;                   >
```

```
<!ELEMENT addName       - -  (%phrase.seq;)                    >
<!ATTLIST addName            %a.global;
                             %a.personPart;                    >
<!ELEMENT roleName      - -  (%phrase.seq)                     >
<!ATTLIST roleName           %a.global;
                             %a.personPart;                    >
<!-- This fragment is used in sec. 20                         -->
```

20.2 Place Names

Like other proper nouns or noun phrases used as names, place names can simply be marked up with the <rs> element, or with the <name> element.For cartographers and historical geographers, however, the component parts of a place name provide important information about the relation between the name and some spot in space and time. They also provide importance evidence in historical linguistics. For such applications and others in which the internal structure of a place name is to be encoded, the <placeName» element and its subcomponents should be used.

<placeName> contains an absolute or relative place name.

<settlement> contains the name of the smallest component of a place name expressed as a hierarchy of geo-political or
administrative units as in "Rochester", New York;
"Glasgow", Scotland.

<region> in an address, contains the state, province, county or region name; in a place name given as a hierarchy of geo-political units, the <region> is larger or administratively superior to the <settlement> and
smaller or administratively less important than the
<country>.

<country> in an address, gives the name of the nation, country, colony, or commonwealth; in a place name given as a hierarchy of geo-political units, the <country> is larger or administratively superior to the <region> and smaller than the <bloc>.

<bloc> a geo-political unit containing one or more nation states.

<geogName> a name associated with some geographical feature such as
"Windrush Valley" or "Mount Sinai".

<geog> contains a common noun identifying some geographical feature contained within a geographic name, such as "valley", "mount" etc.

<distance> that part of a relative temporal or spatial expression which indicates the distance between the place or time denoted by it and the place or time referred to within it.

<offset> that part of a relative temporal or spatial expression
which indicates the direction of the offset between the two place names, dates, or times involved in the expression.

As members of the *names* class, all these elements share the following attributes:

key provides an alternative identifier for the object being named, such as a database record key.

reg gives a normalized or regularized form of the name used.

Additionally, all of the above elements are members of the class *placePart*, and thus share the following attributes:

type provides more culture- linguistic- or application- specific information used to categorize this name component.

full indicates whether the place name component is given in full, as an abbreviation or simply as an initial Legal values are:

yes the name component is spelled out in full.
abb the name component is given in an abbreviated form.
init the name component is indicated only by one initial.

Like the <persName> element discussed in section 20.1 ('Personal Names') on p. 488, the <placeName> element may be regarded simply as an abbreviation for the tags <name type=place> or <rs type=place>. The following encodings are thus equivalent: [1]

```
After spending some time in our
<rs type=place key=NY1>modern
  <name type=place key=BA1>Babylon</name>
</rs>,
<name type=place key=NY1>New York</name>,
I have proceeded to the
<rs type=place key=PH1>City of Brotherly Love</rs>.

After spending some time in our
<placeName key=NY1>modern
  <placeName key=BA1>Babylon</placeName>
</placeName>,
<placeName key=NY1>New York</placeName>,
I have proceeded to the
<placeName key=PH1>City of Brotherly Love</placeName>.
```

As indicated above, the <placeName> may simply contain a character string and its type attribute may be used to provide a sub-categorization of place names. Alternatively, it may contain more detailed sub components. A place name may be analysed in several different ways: as a geo-political unit, using a hierarchy of descriptive names (see section 20.2.1 ('Geo-political Place Names ') on p. 494); in terms of geographic features such as mountains and rivers (see section 20.2.2 ('Geographic Names ') on p. 495); relative to other place names (see section 20.2.3 ('Relative Place Names ') on p. 495.

20.2.1 Geo-political Place Names

A place name is sometimes given as sequence of geo-political or administrative units, often arranged in ascending sequence according to their size or administrative importance, for example: "Rochester, New York", or as a single such unit, for example "Belgium". The more detailed component elements listed above (<settle> for a settlement, such as a village, town or city; <region> for any administrative unit such as a county, parish or state; <country> for a politically recognized national entity; or <bloc> for any grouping of such entities) have been chosen for their generality of application. They may be tailored more closely to project- and culture-specific needs by specifying appropriate values in their respective type attributes, as in the following example:

```
<placeName key=RNY1>
  <settlement type=city>Rochester</settlement>,
  <region type=state>New York</region>
</placeName>

<placeName key=LSEA1>
  <country type=nation>Laos</country>,
  <bloc type=sub-continent>Southeast Asia</bloc>
</gu>
```

Note that, even in the case where only one of these component place name elements is used, the <placeName> element must still be present.

```
I'd rather be in
<placeName>
<settlement key=RNY1 type=city>Rochester</settlement>
</placeName>
than any other place I know.
```

[1] Strictly, a suitable value such as figurative should be added to the two place names which are presented periphrastically in the second example here, in order to preserve the distinction indicated by the choice of <rs> rather than <name> to encode them in the first version.

20.2.2 Geographic Names

Places may also be named in terms of geographic features such as mountains, lakes or rivers, independently of geo-political units. The <geogName> is provided to mark up such names, as an alternative to the <placeName> element discussed above. It contains a sequence of phrase level elements, optionally extended by the following special element:

<geog> contains a common noun identifying some geographical feature contained within a geographic name, such as "valley", "mount" etc.

For example:

```
<geogName key=MIRI1 type=river>
  Mississippi River
</geogName>
```

Where the <geog> element is used to characterize the kind of geographic feature being named, the <name> element will generally also be used to mark the associated proper noun or noun phrase:

```
<geogName key=MIRI1 type=river>
  <name>Mississippi</name>
  <geog>River</geog>
</geogName>
```

A more complex example, showing a variety of practices, follows:

```
The isolated ridge separates two great corridors which run from
<name key=GLCO1 type=place>Glencoe</name> into
<geogName key=GLET1 type=glen>
  <geog reg='glen'>Glen</gfn>
  <name>Etive</name>
</geogName>, the
<geogName key=LAGA1 type=hill>
  <geog lang='gaelic' reg='sloping hill face'>Lairig</gfn>
  <name type=>Gartain</name>
</geogName> and the
<geogName key=LAEI1 type=hill>
  <geog lang='gaelic' reg='sloping hill face'>Lairig</gfn>
  <name>Eilde</name>
</geogName>
```

20.2.3 Relative Place Names

All the place name specifications so far discussed are *absolute*, in the sense that they define only one place. A place may however be specified in terms of its relationship to another place, for example "10 miles northeast of Paris" or "near the top of Mount Sinai". These *relative place names* will contain a place name which acts as a referent (e.g. "Paris" and "Mount Sinai"). They will also contain a word or phrase indicating the the position of the place being named in relation to the referent (e.g. "the top of", "north of"). A distance, possibly only vaguely specified, between the referent place and the place being indicated may also be present (e.g. "10 miles", "near")

Relative place names may be encoded using the following elements in combination with either a <placeName> or a <geogName> element.

<offset> that part of a relative temporal or spatial expression which indicates the direction of the offset between the two place names, dates, or times involved in the expression.

<distance> that part of a relative temporal or spatial expression which indicates the distance between the place or time denoted by it and the place or time referred to within it.

Some examples of relative place names are:

```
<placeName key=NRPA1>
  <offset>near</offset>
  <geog>
```

```
   the top of
   <geogName>Mount</geogName>
   <name>Sinai</name>
  </geog>
</placeName>

<placeName key=NEPA1>
  <distance>10 miles</distance>
  <offset>north of</offset>
  <settlement type=city>Paris
  </settlement>
</placeName>
```

The internal structure of place names is like that of personal names - complex and subject to an enormous amount of variation across time and different cultures. The recommendations in this section will be adequate for a majority of users and applications. They may not, however, satisfy the most specialized inquiries and/or applications in which case it is recommended that the internal structure of place names be represented using feature structures 16 ('Feature Structures') on p. 397.

The elements discussed in this section are formally defined as follows:

```
<!-- 20.2.3:  Names for places                                    -->
<!ELEMENT placeName     - -  ((%m.placePart; | %m.phrase; |
                              #PCDATA)+)                          >
<!ATTLIST placeName          %a.global;
          type               CDATA               #IMPLIED
          full               (yes | abb | init)  yes
                             %a.names;                           >
<!ELEMENT settlement    - -  (%phrase.seq;)                      >
<!ATTLIST settlement         %a.global;
                             %a.placePart;                       >
<!ELEMENT region        - -  (%paraContent)                      >
<!ATTLIST region             %a.global;
                             %a.placePart;                       >
<!ELEMENT country       - o  (%paraContent)                      >
<!ATTLIST country            %a.global;
                             %a.placePart;                       >
<!ELEMENT bloc          - -  (%phrase.seq)                       >
<!ATTLIST bloc               %a.global;
                             %a.placePart;                       >
<!ELEMENT offset        - -  (#PCDATA)                           >
<!ATTLIST offset             %a.global;
          value              CDATA               #IMPLIED
                             %a.placePart;                       >
<!ELEMENT distance      - -  (%phrase.seq)                       >
<!ATTLIST distance           %a.global;
          value              CDATA               #IMPLIED
          type               CDATA               #IMPLIED
          full               (yes | abb | init)  yes
          exact              (Y | N | U)         U
          reg                CDATA               #IMPLIED        >
<!ELEMENT geogName      - -  (geog | name | #PCDATA)*            >
<!ATTLIST geogName           %a.global;
                             %a.placePart;                       >
<!ELEMENT geog          - -  (#PCDATA)                           >
<!ATTLIST geog               %a.global;
                             %a.placePart;                       >
<!-- This fragment is used in sec. 20                            -->
```

20.3 Organization Names

Like names of persons or places, organization names can be marked as referent strings or as proper names with the <rs> and <name> elements. For certain applications it may be desirable to mark the component parts of an organization. In some historical and social scientific studies, for example, the component parts of an organization names may give crucial clues which help to characterizing the organization in terms of its geographical location, ownership, likely number of employees, management structure etc. The elements discussed in this section are recommended for this purpose and include:

<orgName> contains an organizational name. Attributes include:

- **type** more fully describes the organization indicated in the organizational name. Possible values include "voluntary", "political", "governmental", "industrial", "commercial", etc.
- **key** provides an alternative identifier for the organization being named, such as a database record key.
- **reg** gives a normalized or regularized form of the organization name

<orgtitle> contains the proper name component of an organizational name. Attributes include:

- **type** more fully describes the organization title. Possible values include "formal", "colloquial", "acronym", etc.
- **reg** gives a normalized or regularized form of the organization title.

<orgtype> indicates a part of the organization name which contains information about the organization's structure or function. Attributes include:

- **type** more fully describes the organization type specified in the name component. Possible values include "function", "structure", etc.
- **reg** gives a normalized or regularized form of the organization type

<orgdivn> indicates a division, branch or department specified in an organizational name. Attributes include:

- **type** more fully describes the organization division specified in the name component. Possible values include "branch", "department", "section", "division", etc.
- **reg** gives a normalized or regularized form of the organizational division.

The <orgname> element should be used when it is desirable to mark an organization name irrespective of whether or not its components are also to be marked. In effect the <orgname> element is a special case of a <name> and thus of an <rs> element. Consequently, the following examples are synonymous, though the last is preferred:

```
About a year back, a question of considerable interest was
agitated in the <rs type=org key=PAS1>Pennsyla. Abolition Society</rs>.

About a year back, a question of considerable interest was
agitated in the <rs type=org key=PAS1><name>Pennsyla. Abolition
Society</name></rs>.

About a year back, a question of considerable interest
was agitated in the <name type=org key=PAS1>Pennsyla.
Abolition Society</name>.

About a year back, a question of considerable interest was
agitated in the <orgname key=PAS1 type=voluntary
reg=Pennsylvania Abolition Society'>Pennsyla. Abolition
Society</orgname>.
```

Like the <rs> and <name> elements, the <orgname> element has a key attribute with which an external identifier such as a database key can be assigned to the organization name. It also has a type attribute with which the organization named in the expression can be described, and a reg attribute with which the organization name can be presented in a regularized form.

The <orgtitle> element is used to mark the expression which provides the proper name component of an organization name for example:

```
Mr Frost will be able to earn an extra fee from
<orgname type=media key=BSB1>
  <orgtitle type=acronym>BSkyB</orgtitle>
</orgname>
rather than the
<orgname type=media key=BBC1>
   <orgtitle type=acronym
             reg='British Broadcasting Corporation'>
   BBC</orgtitle>
</orgname>
```

Where personal names are encountered as component parts of an organization's title, as in "Ernst & Young", these may be tagged with the appropriate personal name elements as discussed in 20.1 ('Personal Names') on p. 488. Examples include:

```
<orgname type='accountancy partnership' key=EY1>
<orgtitle>
  <persname><surname>Ernst</surname></persname> &
  <persname><surname>Young</surname></persname>
</orgtitle>
</orgname>
```

Organization names may also contain within them place names which, in some applications, may yield vital clues as to the organization's location and or sphere of influence. These components should be tagged with the approprate place name tags 20.2 ('Place Names') on p. 493. Examples include:

```
A spokesman from
<orgname type=computers key=IBM1>
  <orgtitle reg='International Business Machines'>IBM</orgtitle>
  <country reg='United Kingdom' key=UNKI1>UK</country>
</orgname> said...

The feeling in
<country type=nation key=CAN1>Canada</country>
is one of strong aversion to the
<orgname type=government key=USG1>
  <country type=nation key=US1>United States</country>
  Government
</orgname>, and of predilection for self-government
under the
<orgname type=government
      reg='British monarchy'>English Crown</orgname>
```

The <orgtype> element is used to mark those components of an organizatino name which indicate something about the structure or function of the organization. Examples include:

```
<orgname type='utility company' key=WWPC1>
  <region type=state>Washington</region>
  <orgtype type=function>Water Power</orgtype>
  <orgtype reg='incorporated' type=structure>Inc.</orgtype>
</orgname>

THE TICKET which you will receive herewith has been formed by
the
<orgname type=political reg='Whig party' key=WHI1>
  <orgtitle>Democratic Whig</orgtitle>
  <orgtype type=function>Party</orgtype>
</orgname> after the most careful deliberation,
with a reference to all the great objects of NATIONAL, STATE,
COUNTY and CITY concern, and with a single eye to the
```

```
<emph>Welfare and Best Interests of the Community</emph>.
```

Organizational names may also be specified hierarchically particularly where the named organization is itself a department or a branch of a larger organizational entity. "The Department of Modern History, Glasgow University" is an example. The <orgdivn> element is recommended wherever it is desirable to isolate the independent levels of an organizational hierarchy that are specified in an organization name. Examples include:

```
<orgname type=academic key=DMHGU1>
  <orgdivn type=department>Department of
     <orgtype type=function>Modern History</orgtype>
  </orgdivn>,
  <settle type=city>Glasgow</settle>
  <orgtype type=function>University</orgtype>
</orgname>
```

Although highly flexible, the mechanisms discussed here for marking the components of organization names will not cater for every processing need or organizational name that is encountered. Where the internal structure of organization names is highly complex, where name components are particularly ambiguous, or where it is important to indicate the assumptions made in the evaluation of an organization name, then feature structure notation is recommended 16 ('Feature Structures') on p. 397

The formal declaration of the elements discussed in this section include:

```
<!-- 20.3:  Organization names                                  -->
<!ELEMENT orgName       - -  (%phrase.seq)                       >
<!ATTLIST orgName            %a.global;
          type               CDATA              #IMPLIED
          key                CDATA              #IMPLIED
          reg                CDATA              #IMPLIED       >
<!ELEMENT orgtitle      - -  (%phrase.seq)                       >
<!ATTLIST orgtitle           %a.global;
          type               CDATA              #IMPLIED
          reg                CDATA              #IMPLIED       >
<!ELEMENT orgtype       - -  (%phrase.seq)                       >
<!ATTLIST orgtype            %a.global;
          type               CDATA              #IMPLIED
          reg                CDATA              #IMPLIED       >
<!ELEMENT orgdivn       - -  (%phrase.seq)                       >
<!ATTLIST orgdivn            %a.global;
          type               CDATA              #IMPLIED
          reg                CDATA              #IMPLIED       >
<!-- This fragment is used in sec. 20                           -->
```

20.4 Dates and Time

The following elements for the encoding of dates and times were introduced in section 6.4.4 ('Dates and times') on p. 137:

<date> contains a date in any format. Attributes include:

- **calendar** indicates the system or calendar to which the date belongs.
- **value** gives the value of the date in some standard form, usually yyyy-mm-dd.
- **certainty** indicates the degree of precision to be attributed to the date.

<time> contains a phrase defining a time of day in any format. Attributes include:

- **zone** indicates time zone or place name wherever this is necessary to evaluate a temporal expression.
- **value** gives the value of the time in a standard form.

type indicates something about the type of temporal expression being tagged. Legal values are:

am indicates a temporal expression made on the basis of a twelve-hour clock and referring to a time between midnight and noon.

pm indicates a temporal expression made on the basis of a twelve-hour clock and referring to a time between noon and midnight.

24hour indicates a temporal expression made on the basis of a twenty-four-hour clock.

descriptive indicates a temporal expression made in descriptive terms, e.g. "noon".

While adequate for many applications, these elements do not allow for the representation of the internal structure of expressions indicating dates or times, which may however be of importance for the correct interpretation of such expressions, or for certain kinds of analytic applications. In this section, we introduce the following special-purpose elements, for use when the internal structure of a temporal expression is to be encoded:

<dateStruct> contains an internally structured representation of a date.
<timeStruct> contains an internally structured representation for a time of day.

Two types of temporal expressions are envisaged for dates and times: absolute and relative. An *absolute temporal expression* is composed of a sequence of the following elements, possibly interspersed with character data:

<day> the day component of a structured date.
<week> the week component of a structured date.
<month> the month component of a structured date.
<year> the year component of a date.
<second> the second component of a structured time-expression.
<minute> the minute component of a structured time-expression.
<hour> the hour component of a temporal expression such as
<occasion> a temporal expression (either a date or a time) given in terms of a named occasion such as a holiday, a named time of day, or some notable event.

A *relative temporal expression* describes a date or time with reference to some other (absolute) temporal expression, and thus contains the following elements in addition to those listed above:

<distance> that part of a relative temporal or spatial expression which indicates the distance between the place or time denoted by it and the place or time referred to within it.
<offset> that part of a relative temporal or spatial expression which indicates the direction of the offset between the two place names, dates, or times involved in the expression.

As members of the class *temporalExpr* (temporal expression) these elements all share the following attributes:

value supplies the value of a date or time in a standard form.
type provides any application-, linguistic- or culture-specific classication for the component.
reg gives a normalized or regularized form of the temporal expression.

20.4.1 Absolute Dates and Times

An absolute temporal expression which is a date will contain only a sequence of <day>, <month> <week>, <year> or <occasion> elements, as in the following examples:

```
The university's view of American affairs produced
a stinging attack by Edmund Burke in the Commons
debate of
<dateStruct value='26-10-1775'>
  <day value='26'>26</day>
  <month value='10'>October</month>
  <year value='1775'>1775</year>
</dateStruct>
```

Component elements of a <dateStruct> may be repeated, provided that only a single temporal expression is intended:

```
<dateStruct value='14-05-1993'>
  <day type=name>Friday</day>,
  <day type=number>14</day>
  <month>May</month>
  <year>1993</year>
</dateStruct>
```

The <occasion> element may be used for any component of a temporal expression which is given in terms of a named event, such as a public holiday for dates, or a named time such as "tea time" or "matins":

```
In New York,
  <dateStruct value='1-1'>
    <occasion type=holiday>New Years Day</occasion>
  </dateStruct> is the quietest of holidays,
  <dateStruct value='4 July'>
    <occasion type=holiday>Independence Day</occasion>
  </dateStruct>
the most turbulent.
```

These components may be applied to dates using any calendar system using subcomponents equivalent to those listed above:

```
<title>Le Vieux Cordelier:
Journal r&eacute;dig&eacute; par Camille Desmoulins</title>,
<dateStruct type=Revolutionary value='03-02-1794'>
   <day type=name>Quintidi</day>
   <month>Pluviose</month>
   <week value=2>2e d&eacute;cade</week>,
   <year value=2>l'an 2 de la R&eacute;publique Indivisible</year>
</dateStruct>
```

Absolute temporal expressions denoting times which are given in terms of seconds, minutes, hours or of well defined events (e.g. "noon", "sunset") may similarly be represented using the <timeStruct> element:

```
The train leaves for Boston at
  <timeStruct type='24hour' zone=EST>
    <hour>13</hour>:<minute>45</minute>
   </timeStruct>

At <timeStruct>
     <occasion>sunset</occasion>
   </timeStruct> we walked to the beach.

The train leaves for Boston at
  <timeStruct type='descriptive' zone=EST value='13:45'>
     a quarter of <hour reg='1400'>two</hour>
  </timeStruct>
```

The type attribute may be used to distinguish sub-types of component elements (for example, months or days presented as words or as numbers) or to provide additional information about the function of this particular component (for example, to distinguish types of <occasion>). The value and reg attributes are both used to provide a standardized or regularized form of the content of an element. The distinction is that the value specified by the reg attribute is simply that chosen as a convenient way of grouping together a number of variant forms, whereas that specified for the value attribute must always be given in some application-dependent standard form, described in the <stdVals> element of the TEI header.

For example:

```
<dateStruct value='09-06-1807'>
  <month type=name value=06>June</month>
  <day type=number value=09>9th</day>
</dateStruct>:
The period is approaching which will terminate my
```

```
present copartnership. On the
<dateStruct value='01-01-1808'>
  <day type=number value=01>1st</day>
  <month type=name value=01 reg='January'>Jany.</month>
</dateStruct> next, it expires by its own limitation.
```

20.4.2 Relative Dates and Times

As noted above, relative dates and times such as "in the Two Hundredth and First Year of the Republic", "twenty minutes before noon", and, more ambiguously, "after the lamented death of the Doctor" or "an hour after the game" have two distinct components. As well as the absolute temporal expression or event to which reference is made (e.g. "noon", "the game", "the death of the Doctor" "[the foundation of] the Republic"), they also contain a description of the "distance" between the time or date which is indicated and the referent expression (e.g. "the Two Hundredth and First Year", "twenty minutes", "an hour"); and (optionally) an "offset" describing the direction of the distance between the time or date indicated and the referent expression (e.g. "of" implying after, "before", "after").

The elements <distance> (or <measure>) and <offset> are used to encode these last two components within a <dateStruct> or <timeStruct>. The absolute temporal expression contained within the relative expression may be encoded using a <occasion> element, or by a nested <dateStruct> or <timeStruct>, or by a simple <date> or <time>. This allows for endlessly recursive structures such as "the third Sunday after the first Monday before Lammastide in the fifth year of the King's second marriage ..." — but so does natural language.

In the following examples, the reg attribute has been used to simplify processing of variant forms of expression:

```
<dateStruct value='11-12-1786'>
     <distance reg='14 days'>A fortnight</distance
     <offset>before</offset>
     <dateStruct>
       <occasion type='holiday'>Christmas</occasion>
       <year>1786</year>
     </dateStruct>
</dateStruct>

I reached the station
<timeStruct value='14:15'>
 <distance exact=N reg='30 minutes'>
   about a half hour</distance>
 <offset>after</offset>
 <occasion value='13:45'>
   the departure of the afternoon train to Boston</occasion>
</timeStruct>
```

In the following example, the exact attribute has been used to indicate a lack of precision in the distance stated:

```
In practice, festival candles are lit
<timeStruct>
  <distance exact=N>just</distance>
  <offset>before</offset>
  <occasion reg='evening'>sundown</occasion>
</timeStruct>
```

In the following example, a nested <dateStruct> element is used to show that "my birthday" and the cited date are parts of the same temporal expression, and hence to disambiguate the phrase "A week before my birthday on 9th December":

```
<dateStruct value='02-12'>
  <distance>A week</distance>
  <offset>before</offset>
```

```
  <dateStruct value='09-12'>
    <occasion>my birthday</occasion>
    on <day>9th</day><month>December</month>
  </dateStruct>
</dateStruct>
```

The alternative reading of this phrase would be encoded as follows:

```
<dateStruct value='09-12'>
  <distance>A week</distance>
  <offset>before</offset>
  <occasion>my birthday</occasion>
  on <day>9th</day><month>December</month>
</dateStruct>
```

Where more complex or ambiguous expressions are involved, and where it is desirable to make more explicit the interpretive processes required, the feature structure notation described in chapter 16 ('Feature Structures') on p. 397 is recommended. Consider, for example, the following temporal expression which occurs in the *Scottish Temperance Review* of August 1850, referring to the summer holiday known in Glasgow simply as "the Fair":

```
Not only is the city, <date ana=GF50>during the Fair</date>, a
horrible nucleus of immorality and wickedness; it sends our
multitudes to pollute and demoralize the country.
```

For the definition of the `ana` attribute, see chapter 15 ('Simple Analytic Mechanisms') on p. 381. It is used here to link the temporarl phrase with an interpretation of it. Like most traditional fairs and market days, the Glasgow Fair was established by local custom and could vary from year to year. Consequently, in order to provide such an interpretation, it is necessary to drawn upon additional information which may or may not be located in the particular text in question. In this case, it is necessary at least to know the spatial and temporal context (year and place) of the fair referred to. These and other features required for the analysis of this particular temporal expression may be combined together as one feature structure of type `date-analysis`:

```
<fs type=date-analysis id=GF50>
<f name=event><str>the Fair</str></f>
<f name=place><str>Glasgow</str></f>
<f name=year><nbr value=1850></f>
<f name=from-value><sym>08-08-1850</sym></f>
<f name=to-value><sym>19-09-1850</sym></f>
</fs>
```

The elements described in this section are formally defined as follows:

```
<!-- 20.4.2:  Date components                                  -->
<!ELEMENT dateStruct    - -  ((%m.temporalExpr; | #pcdata)+)    >
<!ATTLIST dateStruct         %a.global;
                             %a.temporalExpr;
          calendar           CDATA               #IMPLIED
          exact              CDATA               #IMPLIED      >
<!ELEMENT day           - -  (#PCDATA)                          >
<!ATTLIST day                %a.global;
                             %a.temporalExpr;                  >
<!ELEMENT week          - -  (#PCDATA)                          >
<!ATTLIST week               %a.global;
                             %a.temporalExpr;                  >
<!ELEMENT month         - -  (#PCDATA)                          >
<!ATTLIST month              %a.global;
                             %a.temporalExpr;                  >
<!ELEMENT year          - -  (#PCDATA)                          >
<!ATTLIST year               %a.global;
                             %a.temporalExpr;                  >
<!ELEMENT occasion      - -  (%phrase.seq)                      >
<!ATTLIST occasion           %a.global;
```

```
                                %a.temporalExpr;                    >
<!ELEMENT timeStruct    - -  ((%m.temporalExpr; | #PCDATA)+)    >
<!ATTLIST timeStruct         %a.global;
                             %a.temporalExpr;
          zone               CDATA               #IMPLIED       >
<!ELEMENT second        - -  (#PCDATA)                          >
<!ATTLIST second             %a.global;
                             %a.temporalExpr;                    >
<!ELEMENT minute        - -  (#PCDATA)                          >
<!ATTLIST minute             %a.global;
                             %a.temporalExpr;                    >
<!ELEMENT hour          - -  (#PCDATA)                          >
<!ATTLIST hour               %a.global;
                             %a.temporalExpr;                    >
<!-- offset and distance were defined above                    -->

<!-- This fragment is used in sec. 20                          -->
```

Chapter 21

Graphs, Networks, and Trees

Graphical representations are widely used for displaying relations among informational units because they help readers to visualize those relations and hence to understand them better. Two general types of graphical representations may be distinguished.

- Graphs, in the strictly mathematical sense, consist of points, often called *nodes* or *vertices*, and connections among them, called *arcs*, or under certain conditions, *edges*. Among the various types of graphs are *networks* and *trees*. Graphs generally and networks in particular are dealt with directly below. Trees are dealt with separately in sections 21.2 ('Trees') on p. 514 and 21.3 ('Another Tree Notation') on p. 517.[1]
- *Charts*, which typically plot data in two or more dimensions, including plots with orthogonal or radial axes, bar charts, pie charts, and the like. These can be described using the elements defined in the additional tag set for figures and graphics; see chapter 22 ('Tables, Formulae, and Graphics') on p. 523.

The following DTD fragment shows the overall organization of the tag set discussed in the remainder of this chapter.

```
<!-- 21:  Graphs, networks and trees                    -->
<!-- ... declarations from section 21.1                 -->
<!--     (Graphs)                                       -->
<!--     go here ...                                    -->
<!-- ... declarations from section 21.2                 -->
<!--     (Trees (basic method))                         -->
<!--     go here ...                                    -->
<!-- ... declarations from section 21.3                 -->
<!--     (Trees (alternate method))                     -->
<!--     go here ...                                    -->
```

This tag set is made available as described in 3.3 ('Invocation of the TEI DTD') on p. 39; in a document which uses the markup described in this chapter, the document type declaration should contain the following declaration for the entity *tei.nets*:

```
<!ENTITY % tei.nets 'INCLUDE'>
```

The entire document type declaration for a document using this additional tag set together with the prose base might look like this:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' [
   <!ENTITY % tei.prose 'INCLUDE' >
   <!ENTITY % tei.nets  'INCLUDE' >
]>
```

Among the types of qualitative relations often represented by graphs are organizational hierarchies, flow charts, genealogies, semantic networks, transition networks, grammatical relations, tournament schedules, seating plans, and directions to people's houses. In developing

[1] The treatment here is largely based on the characterizations of graph types in Gary Chartrand and Linda Lesniak, *Graphs and Digraphs* (Menlo Park, CA: Wadsworth, 1986).

recommendations for the encoding of graphs of various types, we have relied on their formal mathematical definitions and on the most common conventions for representing them visually. However, it must be emphasized that these recommendations do not provide for the full range of possible graphical representations, and deal only partially with questions of design, layout and placement.

21.1 Graphs and Digraphs

Broadly speaking, graphs can be divided into two types: *undirected* and *directed*. An undirected graph is a set of *nodes* (or *vertices*) together with a set of pairs of those vertices, called *arcs* or *edges*. Each node in an arc of an undirected graph is said to be *incident* with that arc, and the two vertices which make up an arc are said to be *adjacent*. An directed graph is like an undirected graph except that the arcs are *ordered pairs* of nodes. In the case of directed graphs, the term *edge* is not used; moreover, each arc in an directed graph is said to be *adjacent from* the node from which the arc emanates, and *adjacent to* the node to which the arc is directed. We use the element **<graph>** to encode graphs as a whole, **<node>** to encode nodes or vertices, and **<arc>** to encode arcs or edges; arcs can also be encoded by attributes on the **<node>** element. These elements have the following descriptions and attributes:

<graph> encodes a graph, which is a collection of nodes, and arcs which connect the nodes. Attributes include:

- **type** describes the type of graph. Suggested values include:
  - ***undirected*** undirected graph
  - ***directed*** directed graph
  - ***transition network*** a directed graph with distinguished initial and final nodes
  - ***transducer*** a transition network with up to two labels on each arc
- **label** gives a label for a graph.
- **order** states the order of the graph, i.e., the number of its nodes.
- **size** states the size of the graph, i.e., the number of its arcs.

<node> encodes a node, a possibly labeled point in a graph. Attributes include:

- **label** gives a label for a node.
- **label2** gives a second label for a node.
- **value** provides the value of a node, which is a feature structure or other analytic element.
- **type** provides a type for a node. Suggested values include:
  - ***initial*** initial node in a transition network
  - ***final*** final node in a transition network
- **adjFrom** gives the IDs of the nodes which are adjacent from the current node.
- **adjTo** gives the IDs of the nodes which are adjacent to the current node.
- **adj** gives the IDs of the nodes which are both adjacent to and adjacent from the current node.
- **inDegree** gives the in degree of the node, the number of nodes which are adjacent from the given node.
- **outDegree** gives the out degree of the node, the number of nodes which are adjacent to the given node.
- **degree** gives the degree of the node, the number of arcs with which the node is incident.

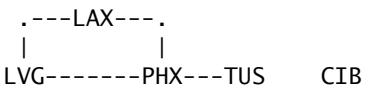
<arc> encodes an arc, the connection from one node to another in a graph. Attributes include:

- **label** gives a label for an arc.
- **label2** gives a second label for an arc.
- **from** gives the ID of the node which is adjacent from this arc.
- **to** gives the ID of the node which is adjacent to this arc.

Before proceeding, some additional terminology may be helpful. We define a *path* in a graph as a sequence of nodes n1, ..., nk such that there is an arc from each ni to ni+1 in the sequence. A *cyclic path*, or *cycle* is a path leading from a particular node back to itself. A graph that contains at least one cycle is said to be *cyclic*; otherwise it is *acyclic*. We say, finally, that a graph is *connected* if

there is a path from some node to every other node in the graph; any graph that is not connected is said to be *disconnected*.

Here is an example of an undirected, cyclic disconnected graph, in which the nodes are annotated with three-letter codes for airports, and the arcs connecting the nodes are represented by horizontal and vertical lines, with 90 degree bends used simply to avoid having to draw diagonal lines.



Airline Connections in Southwestern USA

Next is a markup of the graph, using <arc> elements to encode the arcs.

```
<graph type=undirected
       id=cug1
       label='Airline Connections in Southwestern USA'
       order=5
       size=4>
  <node label=LAX id=LAX degree=2>
  <node label=LVG id=LVG degree=2>
  <node label=PHX id=PHX degree=3>
  <node label=TUS id=TUS degree=1>
  <node label=CIB id=CIB degree=0>
  <arc  from=LAX to=LVG>
  <arc  from=LAX to=PHX>
  <arc  from=LVG to=PHX>
  <arc  from=PHX to=TUS>
  </graph>
```

The label attribute on the <graph> element records a label for the graph; similarly, the label attribute on the <node> elements records the labels of those nodes. The order and size attributes on the <graph> element record the number of nodes and number of arcs in the graph respectively; these values are optional (since they can be computed from the rest of the graph), but if they are supplied, they must be consistent with the rest of the encoding. They can thus be used to help check that the graph has been encoded and transmitted correctly. The degree attribute on the <node> elements record the number of arcs that are incident with that node. It is optional (because redundant), but can be used to help in validity checking: if a value is given, it must be consistent with the rest of the information in the graph. Finally, the from and to attributes on the <arc> elements provide pointers to the nodes connected by those arcs. Since the graph is undirected, no directionality is implied by the use of the from and to attributes; the values of these attributes could be interchanged in each arc without changing the graph.

The adj, adjFrom, and adjTo attributes of the <node> element provide an alternative method of representing unlabeled arcs, their values being pointers to the nodes which are adjacent to or from that node. The adj attribute is to be used for undirected graphs, and the adjFrom and adjTo attributes for directed graphs. It is a semantic error for the directed adjacency attributes to be used in an undirected graph, and vice versa. Here is a markup of the preceding graph, using the adj attribute to represent the arcs.

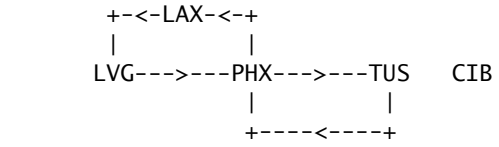
```
<graph type=undirected
       id=cug2
       label='Airline Connections in Southwestern USA'
       order=5
       size=4>
  <node label=LAX id=LAX degree=2 adj='LVG PHX'>
  <node label=LVG id=LVG degree=2 adj='LAX PHX'>
  <node label=PHX id=PHX degree=3 adj='LAX LVG TUS'>
  <node label=TUS id=TUS degree=1 adj='PHX'>
  <node label=CIB id=CIB degree=0>
  </graph>
```

Note that each arc is represented twice in this encoding of the graph. For example, the existence of the arc from LAX to LVG can be inferred from each of the first two <node> elements in the graph. This redundancy, however, is not required: it suffices to describe an arc in any one of the three places it can be described (either adjacent node, or in a separate <arc> element). Here is a less redundant representation of the same graph.

```
<graph type=undirected
       id=cug3
       label='Airline Connections in Southwestern USA'
       order=5
       size=4>
  <node label=LAX id=LAX degree=2 adj='LVG PHX'>
  <node label=LVG id=LVG degree=2 adj='PHX'>
  <node label=PHX id=PHX degree=3 adj='TUS'>
  <node label=TUS id=TUS degree=1>
  <node label=CIB id=CIB degree=0>
  </graph>
```

Although in many cases the <arc> element is redundant (since arcs can be described using the adjacency attributes of their adjacent nodes), it has nevertheless been included in the tag set, in order to allow the convenient specification of identifiers, display or rendition information, and labels for each arc (using the attributes id, rend, and label).

Next, let us modify the preceding graph by adding directionality to the arcs. Specifically, we now think of the arcs as specifying selected routes from one airport to another, as indicated by the direction of the arrowheads in the following diagram.



Selected Airline Routes in Southwestern USA

Here is an encoding of this graph, using the <arc> element to designate the arcs.

```
<graph type=directed
       id=rdg1
       label='Selected Airline Routes in Southwestern USA'
       order=5
       size=5>
   <node label=LAX id=LAX inDegree=1 outDegree=1>
   <node label=LVG id=LVG inDegree=1 outDegree=1>
   <node label=PHX id=PHX inDegree=2 outDegree=2>
   <node label=TUS id=TUS inDegree=1 outDegree=1>
   <node label=CIB id=CIB inDegree=0 outDegree=0>
   <arc  from=LAX to=LVG>
   <arc  from=LVG to=PHX>
   <arc  from=PHX to=LAX>
   <arc  from=PHX to=TUS>
   <arc  from=TUS to=PHX>
   </graph>
```

Here is another encoding of the graph, using the adjTo and adjFrom attributes on nodes to designate the arcs.

```
<graph type=directed
       id=rdg2
       label='Selected Airline Routes in Southwestern USA'
       order=5
       size=5>
   <node label=LAX id=LAX inDegree=1
         outDegree=1 adjTo=LVG adjFrom=PHX>
   <node label=LVG id=LVG inDegree=1
```

```
      outDegree=1 adjFrom=LAX adjTo=PHX>
   <node label=PHX id=PHX inDegree=2
      outDegree=2 adjTo='LAX TUS' adjFrom='LVG TUS'>
   <node label=TUS id=TUS inDegree=1
      outDegree=1 adjTo=PHX adjFrom=PHX>
   <node label=CIB id=CIB inDegree=0 outDegree=0>
   </graph>
```

If we wish to label the arcs, say with flight numbers, then <arc> elements must be used to carry the label attribute, as in the following example.

```
<graph type=directed
       id=rdg1
       label='Selected Airline Routes in Southwestern USA'
       order=5
       size=5>
   <node label=LAX id=LAX >
   <node label=LVG id=LVG >
   <node label=PHX id=PHX >
   <node label=TUS id=TUS >
   <node label=CIB id=CIB >
   <arc  from=LAX to=LVG label='SW117'>
   <arc  from=LVG to=PHX label='SW711'>
   <arc  from=PHX to=LAX label='AA218'>
   <arc  from=PHX to=TUS label='AW229'>
   <arc  from=TUS to=PHX label='AW225'>
   </graph>
```

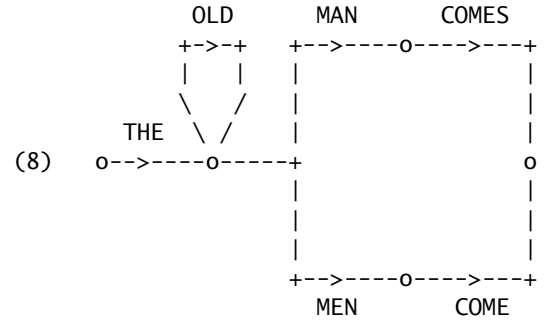
The formal declarations of the <graph>, <node> and <arc> elements are as follows.

```
<!-- 21.1:  Graphs                                          -->
<!ELEMENT graph         - -  ((node)+ & (arc)*)               >
<!ATTLIST graph              %a.global;
          type               CDATA               #IMPLIED
          label              CDATA               #IMPLIED
          order              NUMBER              #IMPLIED
          size               NUMBER              #IMPLIED      >
<!ELEMENT node          - O  EMPTY                             >
<!ATTLIST node               %a.global;
          label              CDATA               #IMPLIED
          label2             CDATA               #IMPLIED
          value              IDREF               #IMPLIED
          type               CDATA               #IMPLIED
          adjTo              IDREFS              #IMPLIED
          adjFrom            IDREFS              #IMPLIED
          adj                IDREFS              #IMPLIED
          inDegree           NUMBER              #IMPLIED
          outDegree          NUMBER              #IMPLIED
          degree             NUMBER              #IMPLIED      >
<!ELEMENT arc           - O  EMPTY                             >
<!ATTLIST arc                %a.global;
          label              CDATA               #IMPLIED
          label2             CDATA               #IMPLIED
          from               IDREF               #REQUIRED
          to                 IDREF               #REQUIRED     >
<!-- This fragment is used in sec. 21                        -->
```

21.1.1 Transition Networks

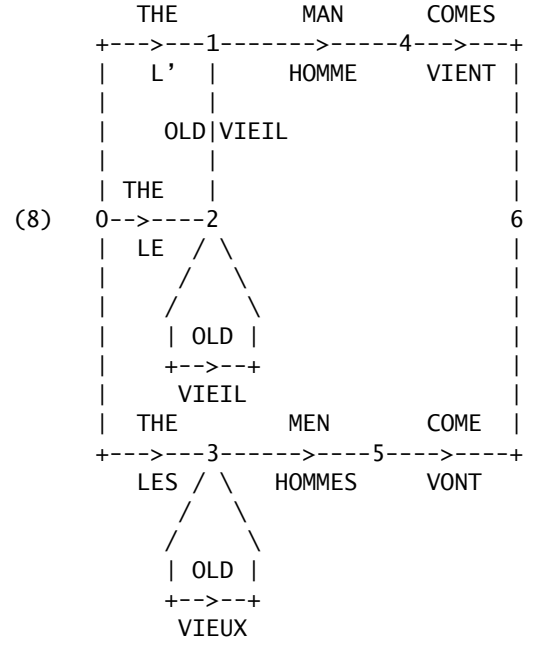
For encoding transition networks and other kinds of directed graphs in which distinctions among types of nodes must be made, the type attribute is provided for <node> elements. In the following example, the *initial* and *final* nodes (or *states*) of the network are distinguished. It can

be understood as accepting the set of strings obtained by traversing it from its initial node to its final node, and concatenating the labels.




```
<graph type='transition network'
       id=ss8
       label='(8)'
       order=5
       size=6>
   <node id=q0 inDegree=0 outDegree=1 type=initial>
   <node id=q1 inDegree=2 outDegree=3>
   <node id=q2 inDegree=1 outDegree=1>
   <node id=q3 inDegree=1 outDegree=1>
   <node id=q4 inDegree=2 outDegree=0 type=final>
   <arc  from=q0 to=q1 label=THE>
   <arc  from=q1 to=q1 label=OLD>
   <arc  from=q1 to=q2 label=MAN>
   <arc  from=q1 to=q3 label=MEN>
   <arc  from=q2 to=q4 label=COMES>
   <arc  from=q3 to=q4 label=COME>
   </graph>
```

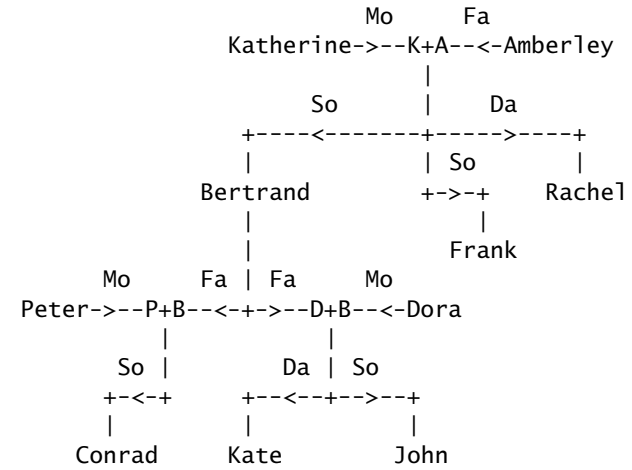
A finite state transducer has two labels on each arc, and can be thought of as representing a mapping from one sequence of labels to the other. The following example represents a transducer for translating the English strings accepted by the network in the preceding example into French. The nodes have been annotated with numbers, for convenience.



```
<graph type='transducer'
       order=7
       size=10>
   <node id=t0 label=0 inDegree=0 outDegree=3 type=initial>
   <node id=t1 label=1 inDegree=2 outDegree=1>
   <node id=t2 label=2 inDegree=2 outDegree=2>
   <node id=t3 label=3 inDegree=2 outDegree=2>
   <node id=t4 label=4 inDegree=1 outDegree=1>
   <node id=t5 label=5 inDegree=1 outDegree=1>
   <node id=t6 label=6 inDegree=2 outDegree=0 type=final>
   <arc  from=t0 to=t1 label=THE label2="L'">
   <arc  from=t0 to=t2 label=THE label2=LE>
   <arc  from=t0 to=t3 label=THE label2=LES>
   <arc  from=t1 to=t4 label=MAN label2=HOMME>
   <arc  from=t2 to=t1 label=OLD label2=VIEIL>
   <arc  from=t2 to=t2 label=OLD label2=VIEIL>
   <arc  from=t3 to=t3 label=OLD label2=VIEUX>
   <arc  from=t3 to=t5 label=MEN label2=HOMMES>
   <arc  from=t4 to=t6 label=COMES label2=VIENT>
   <arc  from=t5 to=t6 label=COME label2=VONT>
   </graph>
```

21.1.2 Family Trees

The next example provides an encoding a portion of a "family tree", in which nodes are used to represent individuals, and parents of individuals, and arcs are used to represent common parentage and descent links. Let us suppose, further, that information about individuals is contained in feature structures, which are contained in feature-structure libraries elsewhere in the document (see 16.3 ('Feature, Feature-Structure and Feature-Value Libraries') on p. 400). We can use the value attribute on <node> elements to point to those feature structures. Assume that, in some particular representation of the graph, nodes representing females are framed by circles, nodes representing males are framed by boxes, and nodes representing parents are framed by diamonds.




```
<graph type='family tree' order=13 size=12>
  <node id=KathR  label='Katherine'
        value=kr1 inDegree=0 outDegree=1>
  <node id=AmbeR  label='Amberley'
        value=ar1 inDegree=0 outDegree=1>
  <node id=KAR  label='K+A'
        inDegree=2 outDegree=3>
  <node id=BertR  label='Bertrand'
        value=br1 inDegree=1 outDegree=2>
  <node id=PeteR  label='Peter'
```

```
      value=pr1 inDegree=0 outDegree=1>
<node id=DoraR  label='Dora'
      value=dr1 inDegree=0 outDegree=1>
<node id=PBR  label='P+B'
      inDegree=2 outDegree=1>
<node id=DBR  label='D+B'
      inDegree=2 outDegree=2>
<node id=FranR  label='Frank'
      value=fr1 inDegree=1 outDegree=0>
<node id=RachR  label='Rachel'
      value=rr1 inDegree=1 outDegree=0>
<node id=ConrR  label='Conrad'
      value=cr1 inDegree=1 outDegree=0>
<node id=KateR  label='Kate'
      value=kr2 inDegree=1 outDegree=0>
<node id=JohnR  label='John'
      value=jr1 inDegree=1 outDegree=0>
<arc label='Mo' from=KathR to=KAR>
<arc label='Fa' from=AmbeR to=KAR>
<arc label='So' from=KAR to=BertR>
<arc label='So' from=KAR to=FranR>
<arc label='Da' from=KAR to=RachR>
<arc label='Mo' from=PeteR to=PBR>
<arc label='Fa' from=BertR to=PBR>
<arc label='So' from=PBR to=ConrR>
<arc label='Mo' from=DoraR to=DBR>
<arc label='Fa' from=BertR to=DBR>
<arc label='Da' from=DBR to=KateR>
<arc label='So' from=DBR to=JohnR>
</grah>
```

21.1.3 Historical Interpretation

For our final example, we represent graphically the relationships among various geographic areas mentioned in a seventeenth-century Scottish document. The document itself is a "sasine", which records a grant of land from the earl of Argyll to one Donald McNeill, and reads in part as follows (abbreviations have been expanded silently, and "[...]" marks illegible passages):

"

Item instrument of Sasine given the said Hector Mcneil confirmed and dated 28 May 1632 [...] at Edinburgh upon the 15 June 1632

Item ane charter granted by Archibald late earl of Argyle and Donald McNeill of Gallachalzie wh makes mention that ... the said late Earl yields and grants to the said Donald MacNeill ...

All and hail the two merk land of old extent of Gallachalzie with the pertinents by and in the lordship of Knapdale within the sherrifdome of Argyll

[description of other lands granted follows ...]

This Charter is dated at Inverary the 15th May 1669"

In this example, we are concerned with the land and pertinents (i.e. accompanying sources of revenue) described as "the two merk land of old extent of Gallachalzie with the pertinents by and in the lordship of Knapdale within the sherrifdom of Argyll".

The passage concerns the following pieces of land:

- the Earl of Argyll's land (i.e. the lands granted by this clause of the sasine)
- two mark of land in Gallachalzie
- the pertinents for this land
- the Lordship of Knapdale
- the sherrifdom of Argyll

We will represent these geographic entities as nodes in a graph. Arcs in the graph will represent the following relationships among them:

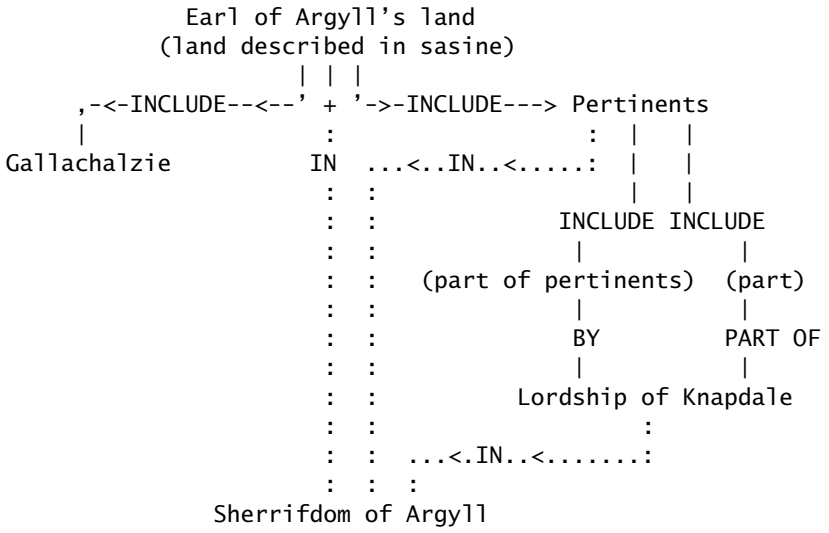
- containment (INCLUDE)

- location within (IN)
- contiguity (BY)
- constituency (PART OF)

Note that these relationships are logically related: "include" and "in", for example, are inverses of each other: the Earl of Argyll's land includes the parcel in Gallachalzie, and the parcel is therefore in the Earl of Argyll's land. Given an explicit set of inference rules, an appropriate application could use the graph we are constructing to infer the logical consequences of the relationships we identify.

Let us assume that feature-structure analyses are available which describe Gallachalzie, Knapdale, and Argyll. We will link to those feature structures using the value attribute on the nodes representing those places. However, there may be some uncertainty as to which noun phrase is modified by the phrase "within the sheriffdome of Argyll": perhaps the entire lands (land and pertinents) are in Argyll, perhaps just the pertinents are, or perhaps only Knapdale is (together with the portion of the pertinents which is in Knapdale). We will represent all three of these interpretations in the graph; they are, however, mutually exclusive, which we represent using the excl attribute defined in chapter 15 ('Simple Analytic Mechanisms') on p. 381.[2]

We represent the graph and its encoding as follows, where the dotted lines in the graph indicate the mutually exclusive arcs; in the encoding, we use the exclude attribute to indicate those arcs.



The graph formalizes the following relationships:

- the Earl of Argyll's land 'includes' (the parcel of land in) Gallachalzie
- the Earl of Argyll's land 'includes' the pertinents of that parcel
- the pertinents are (in part) 'by' the Lordship of Knapdale
- the pertinents are (in part) 'part of' the Lordship of Knapdale
- the Earl of Argyll's land, or the pertinents, or the Lordship of Knapdale, is 'in' the Sherrifdom of Argyll

We encode the graph thus:

```
<graph type=directed order=7 size=9>
  <node id=earl label="Earl of Argyll's land">
  <node id=gall label="Gallachalzie" value=gallfs>
  <node id=pert label="Pertinents">
  <node id=per1 label="Pertinents part">
  <node id=per2 label="Pertinents part">
  <node id=knap label="Lordship of Knapdale" value=knapfs>
```

[2] That is, the three syntactic interpretations of the clause are mutually exclusive. The notion that the pertinents are in Argyll is clearly not inconsistent with the notion that both the land in Gallachalzie and the pertinents are in Argyll. The graph given here describes the possible interpretations of the clause itself, not the sets of inferences derivable from each syntactic interpretation, for which it would be convenient to use the facilities described in chapter 16 ('Feature Structures') on p. 397.

```
<node id=argy label="Sherrifdome of Argyll" value=argyfs>
<arc  id=earlgall label="INCLUDE" from=earl to=gall>
<arc  id=earlargy label="IN" from=earl to=argy
      exclude="pertargy knapargy">
<arc  id=earlpert label="INCLUDE" from=earl to=pert>
<arc  id=pertper1 label="INCLUDE" from=pert to=per1>
<arc  id=pertper2 label="INCLUDE" from=pert to=per2>
<arc  id=pertargy label="IN" from=pert to=argy
      exclude="earlargy knapargy">
<arc  id=per1knap label="BY" from=per1 to=knap>
<arc  id=per2knap label="PART OF" from=per2 to=knap>
<arc  id=knapargy label="IN" from=knap to=argy
      exclude="earlargy pertargy">
</graph>
```

21.2 Trees

A *tree* is a connected acyclic graph. That is, it is possible in a tree graph to follow a path from any vertex to any other vertex, but there are no paths that lead from any vertex to itself. A rooted tree is a directed graph based on a tree; that is, the arcs in the graph correspond to the arcs of a tree such that there is exactly one node, called the *root*, for which there is a path from that node to all other nodes in the graph. For our purposes, we may ignore all trees except for rooted trees, and hence we shall use the **<tree>** element for rooted trees, and the **<root>** element for its root. The nodes adjacent to a given node are called its *children*, and the node adjacent from a given node is called its *parent*. Nodes with both a parent and children are called *internal nodes*, for which we use the **<iNode>** element. A node with no children is tagged as a **<leaf>**. If the children of a node are ordered from left to right, then we say that that node is *ordered*. If all the nodes of a tree are ordered, then we say that the tree is an *ordered tree*. If some of the nodes of a tree are ordered and others are not, then the tree is a *partially ordered tree*. The ordering of nodes and trees may be specified by an attribute; we take the default ordering for trees to be ordered, that roots inherit their ordering from the trees in which they occur, and internal nodes inherit their ordering from their parents. Finally, we permit a node to be specified as following other nodes, which (when its parent is ordered) it would be assumed to precede, giving rise to crossing arcs. The elements used for the encoding of trees have the following descriptions and attributes.

<tree> encodes a tree, which is made up of a root, internal nodes, leaves, and arcs from root to leaves. Attributes include:

- **arity** gives the maximum number of children of the root and internal nodes of the tree.
- **ord** indicates whether or not the tree is ordered, or if it is partially ordered. Legal values are:
  - ***Y*** indicates that all of the branching nodes of the tree are ordered.
  - ***partial*** indicates that some of the branching nodes of the tree are ordered and some are unordered.
  - ***N*** indicates that all of the branching nodes of the tree are unordered.
- **order** gives the order of the tree, i.e., the number of its nodes.

<root> represents the root node of a tree. Attributes include:

- **label** gives a label for a root node.
- **value** provides the value of the root, which is a feature structure or other analytic element.
- **children** provides a list of IDs of the elements which are the children of the root node.
- **ord** indicates whether or not the root is ordered. Legal values are:
  - ***Y*** indicates that the children of the root are ordered.
  - ***N*** indicates that the children of the root are unordered.
- **outDegree** gives the out degree of the root, the number of its children.

<iNode> represents an intermediate (or internal) node of a tree. Attributes include:

- **label** gives a label for an intermediate node.

value provides the value of an intermediate node, which is a feature structure or other analytic element.

children provides a list of IDs of the elements which are the children of the intermediate node.

parent provides the ID of the element which is the parent of this node.

ord indicates whether or not the internal node is ordered. Legal values are:

Y indicates that the children of the intermediate node are ordered.

N indicates that the children of the intermediate node are unordered.

follow provides an ID of the element which this node follows.

outDegree gives the out degree of an intermediate node, the number of its children.

<leaf> encodes the leaves (terminal nodes) of a tree. Attributes include:

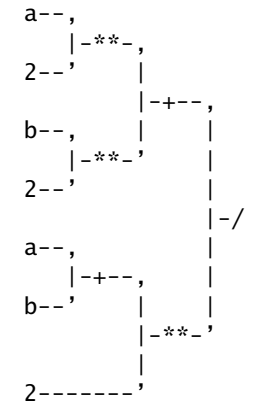
label gives a label for a leaf.

value provides the value of a leaf, which is a feature structure or other analytic element.

parent provides the ID of parent of a leaf.

follow provides an ID of an element which this leaf follows.

Here is an example of a tree. It represents the order in which the operators of addition (symbolized by +), exponentiation (symbolized by **) and division (symbolized by /) are applied in evaluating the arithmetic formula `((a**2)+(b**2))/((a+b)**2)` . In drawing the graph, the root is placed on the far right, and directionality is presumed to be to the left.




```
<tree n='ex1' arity=2 order=12>
   <root  label='/'  id=div1                children='plu1 exp1'>
   <iNode label='+'  id=plu1   parent=div1 children='exp2 exp3'>
   <iNode label='**' id=exp1   parent=div1 children='plu2 num2.3'>
   <iNode label='**' id=exp2   parent=plu1 children='vara1 num2.1'>
   <iNode label='**' id=exp3   parent=plu1 children='varb1 num2.2'>
   <iNode label='+'  id=plu2   parent=exp1 children='vara2 varb2'>
   <leaf  label='a'  id=vara1  parent=exp2>
   <leaf  label='2'  id=num2.1 parent=exp2>
   <leaf  label='b'  id=varb1  parent=exp3>
   <leaf  label='2'  id=num2.2 parent=exp3>
   <leaf  label='a'  id=vara2  parent=plu2>
   <leaf  label='b'  id=varb2  parent=plu2>
   <leaf  label='2'  id=num2.3 parent=exp1>
   </tree>
```

In this encoding, the arity attribute represents the *arity* of the tree, which is the greatest value of the outDegree attribute for any of the nodes in the tree. If, as in this case, arity=2, we say that the tree is a *binary* tree.

Since the left-to-right (or top-to-bottom!) order of the children of the two + nodes does not affect the arithmetic result in this case, we could represent in this tree all of the arithmetically equivalent formulas involving its leaves, by specifying the attribute ord=N on those two <iNode> elements, the attribute ord=Y on the <root> and other <iNode> elements, and the attribute ord=partial on the <tree> element, as follows.

```
<tree n='ex2' ord=partial arity=2 order=13>
 <root  label='/'  id=div1   ord=Y             children='plu1 exp1'>
 <iNode label='+'  id=plu1   ord=N parent=div1 children='exp2 exp3'>
 <iNode label='**' id=exp1   ord=Y parent=div1 children='plu2 num2.3'>
 <iNode label='**' id=exp2   ord=Y parent=plu1 children='vara1 num2.1'>
 <iNode label='**' id=exp3   ord=Y parent=plu1 children='varb1 num2.2'>
 <iNode label='+'  id=plu2   ord=N parent=exp1 children='vara2 varb2'>
 <leaf  label='a'  id=vara1        parent=exp2>
 <leaf  label='2'  id=num2.1       parent=exp2>
 <leaf  label='b'  id=varb1        parent=exp3>
 <leaf  label='2'  id=num2.2       parent=exp3>
 <leaf  label='a'  id=vara2        parent=plu2>
 <leaf  label='b'  id=varb2        parent=plu2>
 <leaf  label='2'  id=num2.3       parent=exp1>
 </tree>
```

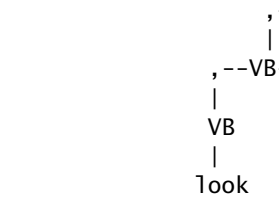
This encoding represents all of the following:

- ((a**2)+(b**2))/((a+b)**2)
- ((b**2)+(a**2))/((a+b)**2)
- ((a**2)+(b**2))/((b+a)**2)
- ((b**2)+(a**2))/((a+b)**2)

Linguistic phrase structure is very commonly represented by trees. Here is an example of phrase structure represented by an ordered tree with its root at the top, and a possible encoding.


```
<tree n='ex3' arity=2 order=8>
   <root  id=PP1               children='P1 NP1'     label='PP'>
   <iNode id=P1    parent=PP1  children=with1        label='P'>
   <leaf  id=with1 parent=P1                         label='with'>
   <iNode id=NP1   parent=PP1  children='the1 peri1' label='NP'>
   <iNode id=Art1  parent=NP1  children=the1         label='Art'>
   <leaf  id=the1  parent=Art1                       label='the'>
   <iNode id=N1    parent=NP1  children=peri1        label='N'>
   <leaf  id=peri1 parent=N1                         label='periscope'>
   </tree>
```

Finally, here is an example of an ordered tree, in which a particular node which ordinarily would precede another is specified as following it. In the drawing, the xxx symbol indicates that the arc from VB to PT crosses the arc from VP to PN.




```
<tree n='ex4' arity=2 order=8>
   <leaf  label='look' id=look1 parent=VB2>
   <leaf  label='them' id=them1 parent=PN1>
   <leaf  label='up'   id=up1   parent=PT1>
   <iNode label='VB'   id=VB2   parent=VB1 children=look1>
   <iNode label='PN'   id=PN1   parent=VP1 children=them1 >
   <iNode label='PT'   id=PT1   parent=VB1 children=up1 follow=PN1>
```

```
<iNode label='VB'   id=VB1   parent=VP1 children='VB2 PT1'>
<root  label='VP'   id=VP1              children='VB1 PN1'>
</tree>
```

The formal declarations of the <tree>, <root>, <iNode> and <leaf> elements are as follows.

```
<!-- 21.2:  Trees (basic method)                                -->
<!ELEMENT tree          - -  ((leaf | iNode)*, root, (leaf |
                             iNode)*)                          >
<!ATTLIST tree               %a.global;
          label              CDATA               #IMPLIED
          arity              NUMBER              #IMPLIED
          ord                (Y | N | partial)   Y
          order              NUMBER              #IMPLIED       >
<!ELEMENT root          - O  EMPTY                             >
<!ATTLIST root               %a.global;
          label              CDATA               #IMPLIED
          value              IDREF               #IMPLIED
          children           IDREFS              #IMPLIED
          ord                (Y | N)             #IMPLIED
          outDegree          NUMBER              #IMPLIED       >
<!ELEMENT iNode         - O  EMPTY                             >
<!ATTLIST iNode              %a.global;
          label              CDATA               #IMPLIED
          value              IDREF               #IMPLIED
          children           IDREFS              #REQUIRED
          parent             IDREF               #IMPLIED
          ord                (Y | N)             #IMPLIED
          follow             IDREF               #IMPLIED
          outDegree          NUMBER              #IMPLIED       >
<!ELEMENT leaf          - O  EMPTY                             >
<!ATTLIST leaf               %a.global;
          label              CDATA               #IMPLIED
          value              IDREF               #IMPLIED
          parent             IDREF               #IMPLIED
          follow             IDREF               #IMPLIED       >
<!-- This fragment is used in sec. 21                          -->
```

21.3 Another Tree Notation

In this section, we present an alternative to the method of representing the structure of ordered rooted trees that is given in section 21.2 ('Trees') on p. 514, which is based on the observation that any node of such a tree can be thought of as the root of the subtree that it dominates. Thus subtrees can be thought of as the same type as the trees they are embedded in, hence the designation <eTree>, for *embedding tree*. Whereas in a <tree>, the relationship among the parts is indicated by the children attribute, and by the names of the elements <root>, <iNode> and <leaf>, the relationship among the parts of an <eTree> is indicated simply by the arrangement of their content. However, we have chosen to enable encoders to distinguish the terminal elements of an <eTree> by means of the empty <eLeaf> element, though its use is not required; the <eTree> element can also be used to identify the terminal nodes of <eTree> elements. We also provide a <triangle> element, which can be thought of as an *underspecified* <eTree>, that is an <eTree> in which certain information has been left out. In addition, we provide a <forest> element, which consists of one or more <tree>, <eTree> or <triangle> elements, and a <forestGrp> element, which consists of one or more <forest> elements. The elements used for the encoding of embedding trees and the units containing them have the following descriptions and attributes.

<eTree> provides an alternative to <tree> element for representing ordered rooted tree

structures. Attributes include:

label gives a label for an embedding tree.

value provides the value of an embedding tree, which is a feature structure or other analytic element.

<triangle> provides for an underspecified <eTree>, that is, an <eTree> with information left out. Attributes include:

label gives a label for an underspecified embedding tree.

value provides the value of a triangle, which is the SGML identifier of a feature structure or other analytic element.

<eLeaf> provides explicitly for a leaf of an embedding tree, which may also be encoded with the <eTree> element. Attributes include:

label gives a label for a leaf of an embedding tree.

value provides the value of an embedding leaf, which is a feature structure or other analytic element.

<forest> provides for groups of rooted trees. Attributes include:

type identifies the type of the forest.

<forestGrp> provides for groups of forests. Attributes include:

type identifies the type of the forest group.

Like the <root>, <iNode> and <leaf> of a <tree>, the <eTree>, <triangle> and <eLeaf> elements may also have label and value attributes.

To illustrate the use of the <eTree> and <eLeaf> elements, here is an encoding of the second example in section 21.2 ('Trees') on p. 514, repeated here for convenience.

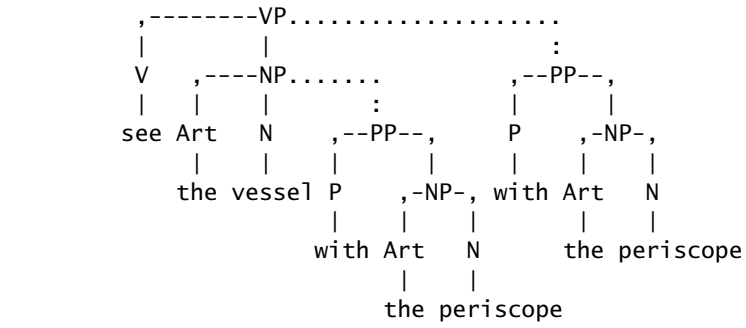

```
<eTree n='ex1' label='PP'>
   <eTree label='P'><eLeaf label='with'></eTree>
   <eTree label='NP'>
      <eTree label='Art'><eLeaf label='the'></eTree>
      <eTree label='N'><eLeaf label='periscope'></eTree>
      </eTree>
   </eTree>
```

Next, we provide an encoding, using the <triangle> element, in which the internal structure of the <eTree> labeled NP is omitted.


```
<eTree n='ex2' label='PP'>
   <eTree label='P'><eLeaf label='with'></eTree>
   <triangle label='NP'><eLeaf label='the periscope'></triangle>
   </eTree>
```

Ambiguity involving alternative tree structures associated with the same terminal sequence can be encoded relatively conveniently using a combination of the exclude and copyOf attributes described in sections 14.8 ('Alternation') on p. 373 and 14.6 ('Identical Elements and Virtual Copies') on p. 367. In the simplest case, an <eTree> may be part of the content of

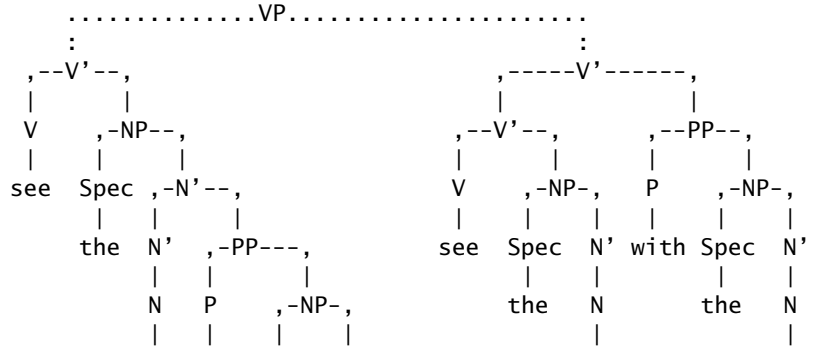
exactly one of two different <eTree> elements. To mark it up, the embedded <eTree> may be fully specified within one of the embedding <eTree> elements to which it may belong, and a virtual copy, specified by the copyOf attribute, may appear on the other. In addition, each of the embedded elements in question is specified as excluding the other, using the exclude attribute. To illustrate, consider the English phrase 'see the vessel with the periscope', which may be considered to be structurally ambiguous, depending on whether the phrase 'with the periscope' is a modifier of the phrase 'the vessel' or a modifier of the phrase 'see the vessel'. This ambiguity is indicated in the sketch of the ambiguous tree by means of the dotted-line arcs. The markup using the copyOf and exclude attributes follows the sketch.




```
<eTree n='ex3' label='VP'>
   <eTree label='V'><eLeaf label='see'></eTree>
   <eTree label='NP'>
      <eTree label='Art'><eLeaf label='the'></eTree>
      <eTree label='N'><eLeaf label='vessel'></eTree>
      <eTree id=ppa exclude=ppb label='PP'>
         <eTree label='P'><eLeaf label='with'></eTree>
         <eTree label='NP'>
            <eTree label='Art'><eLeaf label='the'></eTree>
            <eTree label='N'><eLeaf label='periscope'></eTree>
            </eTree>
         </eTree>
      </eTree>
   <eTree id=ppb copyOf=ppa exclude=ppa label='PP'></eTree>
   </eTree>
```

To indicate that one of the alternatives is selected, one may specify the select attribute on the highest <eTree> as either select=ppa or select=ppb; see section 14.8 ('Alternation') on p. 373.

Depending on the grammar one uses to associate structures with examples like 'see the man with the periscope', the representations may be more complicated than this. For example, adopting a version of the *X-bar* theory of phrase structure originated by Jackendoff[3], the attachment of a modifier may require the creation of an intermediate node which is not required when the attachment is not made, as shown in the following diagram. A possible encoding of this ambiguous structure immediately follows the diagram.



[3]R. Jackendoff, *X-Bar Syntax*, 1977

```
vessel with Spec  N'            vessel        periscope
                   |    |
                  the   N
                        |
                     periscope
```

```
<eTree n='ex4' label="VP">
   <eTree id=vbara exclude=vbarb label="V'">
      <eTree id=va label="V"><eLeaf label="see"></eTree>
      <eTree label="NP">
         <eTree id=spec1a label="Spec"><eLeaf label="the"></eTree>
         <eTree label="N'">
            <eTree id=nbar2a label="N'">
               <eTree label="N"><eLeaf label="vessel"></eTree>
               </eTree>
            <eTree id=ppa label="PP">
               <eTree label="P"><eLeaf label="with"></eTree>
               <eTree label="NP">
                  <eTree label="Spec"><eLeaf label="the"></eTree>
                  <eTree label="N'">
                     <eTree label="N"><eLeaf label="periscope"></eTree>
                     </eTree>
                  </eTree>
               </eTree>
            </eTree>
         </eTree>
   <eTree id=vbara exclude=vbarb label="V'">
      <eTree label="V'">
         <eTree id=vb copyOf=va label="V"></eTree>
         <eTree label="NP">
            <eTree id=spec1b copyOf=spec1a label="Spec"></eTree>
            <eTree id=nbar2b copyOf=nbar2a label="N'"></eTree>
            </eTree>
         </eTree>
      <eTree id=ppb copyOf=ppa label="PP"></eTree>
      </eTree>
   </eTree>
```

A *derivation* in a generative grammar is often thought of as a set of trees. To encode such a derivation, one may use the <forest> element, in which the trees may be marked up using the <tree>, the <eTree> or the <triangle> element. The type attribute may be used to specify what kind of derivation it is. Here is an example of a two-tree forest, involving application of the "wh-movement" transformation in the derivation of 'what you do' (as in 'this is what you do') from the underlying 'you do what '[4].

```
   ,--S'--,                  ,--S'--,
   |      |                  |      |
  COMP ,--S--,              COMP ,--S--,
   |   |     |               |   |     |
   e   NP  ,-VP-,            NP  NP  ,-VP-,
       |   |    |            |   |   |    |
      you  V    NP         what you  V    NP
           |    |                    |    |
           do   what                 do   t
```

```
<forest n='ex5' type='syntactic derivation'>
   <eTree n='Stage 1' id=s1sbar label="S'">
      <eTree id=s1comp label="COMP"><eleaf id=s1e label="e"></eTree>
```

[4]The symbols e and t denote special theoretical constructs (*empty category* and *trace* respectively), which need not concern us here.

```
      <eTree id=s1s label="S">
         <eTree id=s1np1 label="NP"><eLeaf label="you"></eTree>
         <eTree id=s1vp label="VP">
            <eTree id=s1v label="V"><eLeaf label="do"></eTree>
            <eTree id=s1np2 label="NP">
               <eLeaf id=s1wh label="what">
               </eTree>
            </eTree>
         </eTree>
      </eTree>
   <eTree n='Stage 2' id=s2sbar corresp=s1sbar label="S'">
      <eTree id=s2comp corresp=s1comp label="COMP">
         <eTree copyOf=s1np2 corresp=s1e label="NP"></eTree>
         </eTree>
      <eTree id=s2s corresp=s1s label="S">
         <eTree id=s2np1 copyOf=s1np1 label="NP"></eTree>
         <eTree id=s2vp corresp=s1vp label="VP">
            <eTree id=s2v copyOf=s1v label="V"></eTree>
            <eTree id=s2np2 corresp=s1np2 label="NP">
              <eLeaf corresp=s1wh label="t">
              </eTree>
            </eTree>
         </eTree>
      </eTree>
   </forest>
```

In this markup, we have used copyOf attributes to provide virtual copies of elements in the tree representing the second stage of the derivation that also occur in the first stage, and the corresp attribute (see section 14.4 ('Correspondence and Alignment') on p. 358) to link those elements in the second stage with corresponding elements in the first stage that are not copies of them.

If a group of forests (e.g. a full grammatical derivation including syntactic, semantic and phonological subderivations) is to be articulated, the grouping element <forestGrp> may be used.

The formal declarations of the <eTree>, <triangle>, <eLeaf>, <forest> and <forestGrp> elements are as follows.

```
<!-- 21.3:  Trees (alternate method)                          -->
<!ELEMENT eTree         - -  ((eTree | triangle | eLeaf )*)     >
<!ATTLIST eTree              %a.global;
          label              CDATA               #IMPLIED
          value              IDREF               #IMPLIED       >
<!ELEMENT triangle      - -  ((eTree | triangle | eLeaf)*)      >
<!ATTLIST triangle           %a.global;
          label              CDATA               #IMPLIED
          value              IDREF               #IMPLIED       >
<!ELEMENT eLeaf         - O  EMPTY                              >
<!ATTLIST eLeaf              %a.global;
          label              CDATA               #IMPLIED
          value              IDREF               #IMPLIED       >
<!ELEMENT forest        - -  ((tree | eTree | triangle)+)       >
<!ATTLIST forest             %a.global;
          type               CDATA               #IMPLIED       >
<!ELEMENT forestGrp     - -  ((forest)+)                        >
<!ATTLIST forestGrp          %a.global;
          type               CDATA               #IMPLIED       >
<!-- This fragment is used in sec. 21                          -->
```

Chapter 22

Tables, Formulae, and Graphics

Many documents, both historical and contemporary, include not only text but also graphics, artwork, and other images. Although such images could be represented directly within SGML, SGML is not primarily designed for that purpose, and it is standard practice to include such information in SGML documents by declaring each image as an external entity encoded in a suitable non-SGML notation, and then referring to it from within the document. In addition to graphic images, documents often contain material presented in graphical or tabular format. In such materials, details of layout and presentation may also be of comparatively greater significance or complexity than they are for running text. Indeed, it may often be difficult to make a clear distinction between details relating purely to the rendition of information and those relating to the information itself.

Finally, documents may contain mathematical formulæ or expressions in other formulaic notations, for which no SGML notation is defined in these Guidelines.

These areas (graphics, tabular material, and mathematical or other formulæ) have in common that they have received considerable attention from many other standards bodies or similar professional groups. In part because of this, they may frequently be most conveniently encoded and processed using some non-SGML notation, or some SGML notation not defined by these Guidelines. For these reasons, and others, we consider tables, formulæ, and graphics together in this chapter.

As with text markup in general, many incompatible formats have been proposed for the representation of graphics, formulae and tables in electronic form. Unfortunately, no single format as effective as SGML in the domain of text has yet emerged for their interchange, to some extent because of the difficulty of representing the information these data formats convey independently of the way it is rendered.

The additional tag set defined by this chapter defines special purpose "container" elements that can be used to encapsulate occurrences of such data within a TEI-conformant document in a portable way. Specific recommendations for the encoding of tables are provided in section 22.1 ('Tables') on p. 524 and recommendations for mathematical or other formulæ in section 22.2 ('Formulae') on p. 527. Specific recommendations for the encoding of graphic figures may be found in section 22.3 ('Specific Elements for Graphic Images') on p. 530. The rest of the chapter is devoted to general problems of encoding graphic information.

There is at the time of writing no consensus on formats for graphical images, and such formats vary in many ways. We therefore provide (in section 22.4 ('Overview of Basic Graphics Concepts') on p. 532) a brief discussion of the ways in which images may be represented, and (in section 22.5 ('Graphic Image Formats') on p. 533) a list of formal names for those representations most popular at this time. Each one includes a very brief description and (where known) a reference to the formal specification of the notation. These Guidelines recommend a few particular representations as being the most widely supported and understood.

To enable the additional tag set defined by this chapter, the parameter entity *TEI.figures* must be defined with the value `INCLUDE`, as shown in the example below:

```
<!DOCTYPE tei.2 system 'tei2.dtd' [
   <!ENTITY % TEI.figures 'INCLUDE' >
```

```
]>
```

With this declaration in effect, the TEI elements and attributes described in the following sections are all available. If any of the specialized notations described in sections 22.2 ('Formulae') on p. 527 and 22.3 ('Specific Elements for Graphic Images') on p. 530 are used, then an additional *notation declaration* must also be included in the document type declaration subset, as further illustrated below.

The overall structure of the tag set defined in this chapter is as follows:

```
<!-- 22:  Tables, Formulae, Figures                    -->
<!-- ... declarations from section 22.1.1              -->
<!--     (Tables)                                      -->
<!--     go here ...                                   -->
<!-- ... declarations from section 22.2                -->
<!--     (Formulae)                                    -->
<!--     go here ...                                   -->
<!-- ... declarations from section 22.3                -->
<!--     (Figures)                                     -->
<!--     go here ...                                   -->
```

22.1 Tables

A table is the least "graphic" of the elements discussed in this chapter. Almost any text structure can be presented as a series of rows and columns: one might, for example, choose to show a glossary or other form of list in tabular form, without necessarily regarding it as a table. In such cases, the global **rend** attribute is an appropriate way of indicating that some element is being presented in tabular format. When tabular presentation is regarded as of less intrinsic importance, it is correspondingly simpler to encode descriptive or functional information about the contents of the table, for example to identify one column as containing names and another as containing dates, though the two methods may be combined.

When, however, particular SGML elements are required to encode the tabular arrangement itself, then one or other of the various "table DTDs" now available may be preferable. The table DTDs in common use generally view a table as a special text element, made up of row elements (or, sometimes, column elements), themselves composed of cells. Table cells generally appear in row-major order, with the first row from left to right, then the second row, and so on. Details of appearance such as column widths, border lines, and alignment are generally encoded by numerous attributes. Beyond this, however, such DTDs differ greatly. This section begins by describing a table DTD of this kind; a brief summary of some other widely available table DTDs is also provided in section 22.1.2 ('Other Table DTDs') on p. 527.

22.1.1 The TEI Table DTD

For encoding tables of low to moderate complexity, these Guidelines provide the following special purpose elements:

<table> contains text displayed in tabular form, in rows and columns. Attributes include:
- **rows** indicates the number of rows in the table.
- **cols** indicates the number of columns in each row of the table.

<row> contains one row of a table. Attributes include:
- **role** indicates the kind of information held in the cells of this row. Suggested values include:
  - ***label*** labelling or descriptive information only.
  - ***data*** data values.

<cell> contains one cell of a table. Attributes include:
- **role** indicates the kind of information held in the cell. Suggested values include:
  - ***label*** labelling or descriptive information only.
  - ***data*** data values.

cols indicates the number of columns occupied by this cell.
rows indicates the number of rows occupied by this cell.

The <table> element is defined as a member of the class *inter*; it may therefore appear both within other components (such as paragraphs), or between them, provided that the additional tag set defined in this chapter has been enabled, as described at the beginning of this chapter.

It is to a large extent arbitrary whether a table should be regarded as a series of rows or as a series of columns. For compatibility with currently available systems, however, these Guidelines require a row-by-row description of a table. It is also possible to describe a table simply as a series of cells; this may be useful for tabular material which is not presented as a simple matrix.

The attributes rows and cols may be used to indicate the size of a table, or to indicate that a particular cell of a table spans more than one row or column. For both tables and cells, rows and columns are always given in top-to-bottom, left-to-right order. These Guidelines do not require that the size of a table be specified; for most formatting and many other applications, it will be necessary to process the whole table in two passes in any case.

Where cells span more than one column or row, the encoder must determine whether this is a purely presentational effect (in which case the rend attribute may be more appropriate), whether the part of the table affected would be better treated as a nested table, or whether to use the spanning attributes listed above.

The role attribute may be used to categorize a single cell, or set a default for all the cells in a given row. The present Guidelines distinguish the roles of *label* and *data* only, but the encoder may define other roles, such as "derived", "numeric", etc., as appropriate.

The following simple example demonstrates how the data presented as a labelled list in section 6.7 ('Lists') on p. 149 might be represented by an encoder wishing to preserve its original appearance as a table:

```
<table rows=2 cols=2 rend=boxed>
<head rend=it>Report of the conduct and progress
of Ernest Pontifex.  Upper Vth form &mdash; half
term ending Midsummer 1851</head>
<row><cell role=label>Classics</cell>
     <cell>Idle listless and unimproving</cell>
</row>
<row><cell role=label>Mathematics</cell>
     <cell>ditto</cell>
</row>
<row><cell role=label>Divinity</cell>
     <cell>ditto</cell>
</row>
<row><cell role=label>Conduct in house</cell>
      <cell>Orderly</cell>
</row>
<row><cell role=label>General conduct</cell>
     <cell>Not satisfactory, on account of his great
     unpunctuality and inattention to duties</cell>
</row></table>
```

Note that this encoding makes no attempt to represent the full significance of the "ditto" cells above; these might be regarded as simple links between the cells containing them and that to which they refer, or as virtual copies of it. For ways of representing either interpretation, see chapter 14 ('Linking, Segmentation, and Alignment') on p. 331.

The following example demonstrates how a simple statistical table may be represented using this scheme:

```
<table rows=4 cols=4>
<head>Poor Man's Lodgings in Norfolk (Mayhew, 1843)
<row role=label>
  <cell>
  <cell>Dossing Cribs or Lodging Houses
  <cell>Beds
  <cell>Needys or Nightly Lodgers
```

```
<row role=data>
  <cell role=label>Bury St Edmund's
  <cell>5<cell>8<cell>128
<row role=data>
  <cell role=label>Thetford
  <cell>3<cell>6<cell>36
<row role=data>
  <cell role=label>Attleboro'
  <cell>3<cell>5<cell>20
<row role=data>
  <cell role=label>Wymondham
  <cell>1<cell>11<cell>22
<!-- ... -->
</table>
```

Note the use of a blank cell in the first row to ensure that the column labels are correctly aligned with the data. Again, this encoding does not explicitly represent the alignment between column and row labels and the data to which they apply. Where the primary emphasis of an encoding is on the semantic content of a table, a more explicit mechanism for the representation of structured information such as that provided by the feature structure mechanism described in chapter 16 ('Feature Structures') on p. 397 may be preferred. Alternatively, the general purpose linkage and alignment mechanisms described in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331 may also be applied to individual cells of a table.

The content of a table cell need not be simply character data. It may also contain any sequence of the phrase level elements described in chapter 6 ('Elements Available in All TEI Documents') on p. 119, thus allowing for the encoding of potentially more useful semantic information, as in the following example, where the fact that one cell contains a number and the other contains a place name has been explicitly recorded:

```
<TABLE>
   <head>US State populations, 1990</head>
   <row role=data>
        <cell><name>Wyoming</>
        <cell><num>453,588</></row>
   <row><cell><name>Alaska</>
        <cell><num>550,043</></row>
   <row><cell><name>Vermont</>
        <cell><num>562,758</></row>
   <row><cell><name>District of Columbia</>
        <cell><num>606,900</></row>
   <row><cell><name>North Dakota</>
        <cell><num>638,800</></row>
   <row><cell><name>Delaware</>
        <cell><num>666,168</></row>
   <row><cell><name>South Dakota</>
        <cell><num>696,004</></row>
   <row><cell><name>Montana</>
        <cell><num>799,065</></row>
   <row><cell><name>Rhode Island</>
        <cell><num>1,003,464</></row>
  <!-- ... -->
</table>
```

In syntactic terms this is little more than a name-change; however, the new names are more useful in that they convey something about the nature and significance of the information, rather than merely suggesting how to display it in rows and columns. The TEI table elements are defined as follows:

```
<!-- 22.1.1:  Tables                                              -->
<!ELEMENT table         - -  (head*, row+)                         >
<!ATTLIST table              %a.global;
          rows               NUMBER              #IMPLIED
```

```
          cols                NUMBER              #IMPLIED       >
<!ELEMENT row           - O   ((cell | table)+)                   >
<!ATTLIST row                 %a.global;
          role                CDATA               data           >
<!ELEMENT cell          - O   (%paraContent)                      >
<!ATTLIST cell                %a.global;
          role                CDATA               data
          rows                NUMBER              1
          cols                NUMBER              1              >
<!-- This fragment is used in sec. 22                           -->
```

22.1.2 Other Table DTDs

Many SGML authoring systems now include built-in support for their own or for public table DTDs. These often provide an enhanced user interface and good formatting capabilities, but are often product-specific, despite their use of SGML.

The SGML DTD developed by the Association of American Publishers (AAP) and standardized in ANSI Z39.59 provided a very simple encoding for correspondingly simple tables. This has been further developed, together with the table DTD documented in ISO Technical Report 9537, and now forms part of ISO 12083. The TEI DTD fragment described above has functionality very similar to that defined by ISO 12083.

For more complex tables, the most effective publically-available DTD is probably that developed by the US Department of Defense CALS project. This supports vertical and horizontal spanning and various kinds of text rotation and justification within cells and is also directly supported by a number of existing SGML software systems.

The CALS table DTD is much too complex to describe fully here; information on it can be obtained, among other places, from the Graphic Communications Association in Alexandria, Virginia. The formal name of the CALS SGML requirements is MIL-M-28001A. Tables conforming to the CALS DTD may be incorporated into documents conforming to these Guidelines, but this may require substantial modification of the TEI DTD which should not be undertaken without expert advice.

22.2 Formulae

Mathematical and chemical formulæ pose similar problems to those posed by tables in that rendition may be of great significance and hard to disentangle from content. They also require access to a wide range of special characters, for most of which standard entity names already exist in the documented ISO entity sets (see further chapters 4 ('Characters and Character Sets') on p. 71 and 25 ('Writing System Declaration') on p. 571).

Formulæ and tables are also similar in that well-researched and detailed SGML DTD fragments have already been developed for them independently of the TEI. They differ in that (for mathematics at least) there also exists a richly detailed text-based but non-SGML notation which is very widely used: this is the TEX system, and the sets of descriptive macros developed for it such as LATEX and AMS-Tex.

The AAP and ISO standards mentioned in section 22.1 ('Tables') on p. 524 above both provide SGML DTDs for equations as well as for tables, which at the time of writing are being updated and unified to form part of ISO 12083. There is also also a third and more general-purpose ongoing mathematical DTD effort known as EuroMath.

As with tables, in all the SGML solutions a tension exists between the need to encode the way a formula is written (its appearance) and the need to represent its semantics. If the object of the SGML encoding is purely to act as an interchange format among different formatting programs, then there is no need to represent the mathematical meaning of an expression. If however the object is to use the SGML encoding as input to an algebraic manipulation system (such as Mathematica or Maple) or a database system, clearly simply representing superscripts and subscripts will be inadequate.

The present Guidelines make no attempt to add to the number of available DTDs for representing formulæ. Instead, we recommend that the user make an informed choice from those already available. The additional tag set described in this chapter makes available only the following element, which should be used to encode any formula, no matter what notation is employed:

<formula> contains a mathematical or other formula. Attributes include:

notation supplies the name of a previously defined notation used for the content of the element.

The legal content of a <formula> is determined by two factors. The parameter entity *formulaContent* supplies an SGML content model for the element; while the notation attribute specifies what SGML *notation* is employed by the element. The default value of the *formulaContent* entity is CDATA, but may be redefined in the document's DTD subset, for example as follows:

```
<!ENTITY % formulaContent "(#PCDATA)">
```

With this declaration in force, formulæ may contain parsed character data, (and may thus use SGML entity references for special symbols and the like). An alternative might be to embed the elements defined by some other tag set, such as that of ISO 12083. [1]

Secondly, when it is necessary to inform an SGML processor that the content of a <formula> should not be treated as ordinary SGML data, because it uses a different notation, a *notation* must be specified. Each notation used by a document must be declared in its DTD subset, as in the following example:

```
<!DOCTYPE tei.2 [
<!ENTITY % TEI.graphics "INCLUDE">
<!NOTATION TeX PUBLIC
  '-//Donald E. Knuth 1983//NOTATION TeX Mathematical Markup//EN'>
<!ENTITY % formulaContent "(#PCDATA)"  >
]>
```

With these declarations in force, a document may include formulæ expressed using standard TEX conventions, as in the following example:

```
<p>Achilles runs ten times faster than the tortoise and
gives the animal a headstart of ten meters.  Achilles runs
those ten meters, the tortoise one; Achilles runs that
meter, the tortoise runs a decimeter; Achilles runs that
decimeter, the tortoise runs a centimeter; Achilles runs
that centimeter, the tortoise, a millimeter; Fleet-footed
Achilles, the millimeter, the tortoise, a tenth of a
millimeter, and so on to infinity, without the tortoise ever
being overtaken. . . Such is the customary version.
<!-- ... -->
The problem does not change, as you can see; but I would
like to know the name of the poet who provided it with a
hero and a tortoise.  To those magical competitors and to
the series
<formula notation=TeX>
$$
{1 \over 10} +
{1 \over 100} +
{1 \over 1000} +
{1 \over 1000} +
{1 \over 10,\!000} +
\dots
$$
</formula>
the argument owes its fame.
```

[1] In this case additional redefinitions may also be needed to avoid name clashes with existing TEI elements. For further details see chapter 29 ('Modifying the TEI DTD') on p. 619.

The notation attribute supplies the name of a defined SGML notation ("LaTeX"), which is associated by its declaration in the DTD subset with an external public entity. How that declaration is resolved will depend on the kind of processor in use, and is outside the scope of these Guidelines. The declaration for the *formulaContent* parameter entity in the DTD will determine how the contents of any <formula> elements will first be parsed by the SGML processor before they are handed to whatever procedure is intended to handle the external notation. When as here it is declared with a value of (#PCDATA), then any SGML entity references in the content will be resolved. This may be useful if the external mechanism uses characters which would otherwise be regarded as significant to the SGML processor: for example, the less-than or greater-than signs. If a less-than sign or another SGML delimiter occurs in a context meaningful to SGML (for the most common delimiters this mean before any letter), the SGML processor will attempt to interpret it: For example, in this expression:

```
<formula notation=tex> a<b </formula>
```

because the less-than sign is followed by a letter, it would be recognized by SGML as beginning a start-tag (for a presumably nonexistent element <b>). One way of avoiding this problem is to represent the less-than character by an entity reference:

```
<formula notation=tex> a&lt;b </formula>
```

Of course, if the notation permits it (as TeX does), a simpler solution to this specific problem is to insert a space after the less-than sign:

```
<formula notation=tex> a < b </formula>
```

Entity references may only appear in the content of a <formula> element when the entity *formulaContent* has been redefined as above. By default, *formulaContent* is defined as CDATA, which means that the only SGML processing carried out is to search for an end-tag. This means that the character sequence </ may not be followed by a > or a letter anywhere within a formula; outside an SGML context, this sequence is fairly unlikely. If it does appear, or if an SGML notation is used, then the parameter entity must be redefined appropriately.

The following SGML-based notations for encoding formulæ are recommended by these Guidelines:

```
<!NOTATION iso12083 PUBLIC
   'ISO 12083:1993//DTD Formulae//EN'>
<!NOTATION euromath PUBLIC
   '-//Euromath 1992//DTD Euromath equations//EN'>
```

In-line versus block placement for an equation can be distinguished if desired, via the global rend attribute. The global n and id attributes may also be used to label or identify the formula, as in the following (imaginary) example:

```
<p>The volume of a sphere is given by the formula:
<formula n=12 rend=inline id=F12 notation=tex>
   $$V = {4\over 3} \pi r^3$$
</formula>
which is readily calculated.
</p>
<p>
As we have seen in equation <ptr target=F12>, ....
```

The *formulaContent* and *formulaNotations* parameter entities are defined as follows:

```
<!-- 22.2:  Formula Content                                  -->
<!ENTITY % formulaNotations 'CDATA'                          >
<!ENTITY % formulaContent 'CDATA'                            >
```

The formula element itself is defined as follows:

```
<!-- 22.2:  Formulae                                         -->
<!ELEMENT formula       - -  %formulaContent;                >
<!ATTLIST formula            %a.global;
          notation           %formulaNotations   #REQUIRED   >
<!-- This fragment is used in sec. 22                        -->
```

22.3 Specific Elements for Graphic Images

The following special purpose elements are provided by this tag set to indicate the presence of graphic images within a document:

<figure> indicates the location of a graphic, illustration, or figure. Attributes include:

entity names the external entity within which the graphic image of the figure is stored.

<figDesc> contains a brief prose description of the appearance or content of a graphic figure, for use when documenting an image without displaying it.

Inclusion of a graphic image in an SGML document of any kind typically requires three distinct steps:

1. The *notation* employed by the image itself must be defined; this is done with an SGML notation declaration in the document type definition.
2. The external entity in which the image is stored must be defined; this is done with an SGML entity declaration, which refers to the notation declared at step one.
3. Within the document, the **<figure>** element is used to mark the position of the image, which is referenced by name, like any other kind of external entity.

In the TEI scheme, these three functions are carried out as follows.

Declarations for all notations used by a document must be provided within the DTD subset, as described above in section 22.2 ('Formulae') on p. 527. Many such notations are in common use; for details see section 22.5 ('Graphic Image Formats') on p. 533.

Entity declarations for the system or public entities containing the graphics themselves must be made within the document's DTD subset, either directly or by including them within a suitable file, as in the example below.

```
<!DOCTYPE tei.2 system 'tei2.dtd'[
<!ENTITY % TEI.prose "INCLUDE">
<!ENTITY % TEI.graphics "INCLUDE">

<!-- Graphics notations used in this document ... -->
<!NOTATION tiff PUBLIC
    '-//Aldus Corp.//NOTATION Tagged Image File Format//EN' >
<!NOTATION cgm PUBLIC
    'ISO 8632-4:1987//NOTATION Clear text encoding//EN' >
<!NOTATION bmp  PUBLIC
    '-//Microsoft, Inc.//NOTATION BMP:  Bitmapped graphic
              format//EN' >


<!-- The file 'figures.ent' contains entity declarations -->
<!-- for all external entities needed by this document -->
<!ENTITY % myFigures "figures.ent"> %myFigures;
]>
```

The file *figures.ent* will contain a series of declarations like the following:

```
<!ENTITY Fig1    system  "fig1.cgm"    NDATA cgm>
<!ENTITY Fig1th  system  "fig1.bmp"    NDATA bmp>
<!ENTITY pullman system  "pullman.tif" NDATA tiff>
```

the effect of which is to associate the name *Fig1* with the system entity *fig1.cgm*, and also to declare that that entity uses the non-SGML notation called "CGM", which is declared in the DTD subset. In the same way, the external entity *fig1.bmp* is defined as using the "BMP" notation, and may be referenced by the name *Fig1th* (see further below).

Finally, the **<figure>** element is used to indicate the location of the graphic image in the text. For example:

```
<figure entity='Fig1'></figure>
```

Note that an end-tag is always required for this element. Three kinds of content may be supplied: the element <head> may be used to transcribe (or supply) a descriptive heading or title for the graphic itself as in this example:

```
<figure entity='Fig1'>
<head>Figure One: The View from the Bridge</head></figure>
```

Figures are often accompanied not only by a title or heading, but by a paragraph or so of commentary or caption. One or more <p> elements following the <head> may be used to transcribe any caption or discussion of the figure in the source:

```
<figure entity=pullman>
<head>Above:</head>
<p>The drawing room of the Pullman house, the white and
gold saloon where the magnate delighted in giving
receptions for several hundred people.</p>
<figDesc>The figure shows an elaborately decorated room,
at least twenty-five feet side to side and fifty feet
long, with ornate mouldings and Corinthian columns
on the walls, overstuffed armchairs and loveseats
arranged in several conversational groupings, and
two large chandeliers.</figDesc>
</figure>
```

Here, the paragraph "The drawing room ... several hundred people" is transcribed from the source, while the description is provided by the encoder, for use by applications which cannot display the graphic directly. In documents created in electronic form with the needs of print-handicapped readers in mind, the <figDesc> element may be provided by the author rather than a subsequent encoder.

```
<figure entity='Fig1'>
<head>Figure One: The View from the Bridge</head>
<figDesc>A Whistleresque view showing four
or five sailing boats in the foreground, and a
series of buoys strung out between them.</figDesc>
</figure>
```

Where the graphic itself contains large amounts of text, perhaps with a complex structure, and perhaps difficult to distinguish from the graphic, the encoder should choose whether to regard the graphic as containing the text (in which case, a nested <text> element may be included within the <figure> element) or to regard the enclosed text as being a separate division of the <text> element in which the graphic appears. In this latter case, an appropriate *divn* class element may be used for the text represented within the graphic, and the <figure> element embedded within it. The choice will depend to a large degree on the encoder's understanding of the relationship between the graphic and the surrounding text.

Like any other element in the TEI scheme, figures may be given identifiers so that they can be aligned with other elements, and linked to or from them, as described in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331. Some common examples are discussed briefly here; full information is provided in that chapter.

It is often desirable to maintain two versions of an image in an electronic file: one a low resolution or "thumbnail" version which, when selected by the user, causes the other, high resolution, version to be accessed. In TEI terms, the thumbnail image acts as a *reference* to the other. Referring to the example above, we will assume that the entity "Fig1th" contains a thumbnail version of the full Fig1 entity. We can now embed a reference to that image using the simple <ref> element discussed in section 6.6 ('Simple Links and Cross References') on p. 147:

```
<ref target=im1>Click here
    <figure entity=fig1th></>
     for enlightenment
</ref>
<!-- ... -->
<!-- elsewhere in the document -->
```

```
<figure id=im1 entity=fig1></>
<!-- other figures here -->
```

Another common requirement is to associate part or the whole of an image with a textual element not necessarily contiguous to it in the text; this is sometimes known as a *callout*. Again, chapter 14 ('Linking, Segmentation, and Alignment') on p. 331 should be consulted for the full details of the mechanisms available for this purpose. This example assumes that we wish to associate one portion of the image held as "fig1" with chapter two of some text, and another portion of it with chapter three. The application may be thought of as a hypertext browser in which the user selects from a graphic image which part of a text to read next, but the mechanism is independent of this particular application.

The first requirement is some way of identifying and hence pointing to sub-parts of a graphic image. This is most easily done using the extended pointer syntax discussed in section 14.2 ('Extended Pointers') on p. 340: thus

```
<xptr id=pd1 doc=Fig1 from='space (0 0 9 9)'>
<xptr id=pd2 doc=Fig1 from='space (40 90 59 119)'>
```

These <xptr> elements identify two areas within the image "Fig1" using the TEI extended pointer syntax. The first (with identifier "pd1") is a square of size 10 by 10, tangent to the origin. The second (with identifier "pd2") is a rectangle of size 20 by 30, starting at the point with co-ordinates (40,90) in the co-ordinate system used by this document.

The next requirement is some way of identifying the parts of the document to which a link is to be made. The most obvious way of doing this is to use the global id attribute:

```
<div1 type=chapter id=C1>
  <!-- text of chapter one here -->
<div1 type=chapter id=C2>
  <!-- text of chapter two here -->
```

Now, all that is needed to linking these areas to the relevant chapters is a <linkGrp> element, as described in section 14.1 ('Pointers') on p. 333:

```
<linkGrp type=callout>
  <link targets='C1 pd1'>
  <link targets='C2 pd2'>
  <!-- ... -->
</linkGrp>
```

Further examples of this technique are provided in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331.

The elements discussed in this section are defined as follows:

```
<!-- 22.3:  Figures                                          -->
<!ELEMENT figure        - -  (head?, p*, figDesc?, text?)     >
<!ATTLIST figure             %a.global;
          entity             ENTITY              #IMPLIED     >
<!ELEMENT figDesc       - -  (%paraContent;)                  >
<!ATTLIST figDesc            %a.global;                       >
<!-- This fragment is used in sec. 22                        -->
```

22.4 Overview of Basic Graphics Concepts

The first major distinction in graphic representation is that between raster graphics and vector graphics. A *raster image* is a list of points, or dots. Scanners, fax machines and other simple devices easily produce digital raster images, and such images are therefore quite common. A *vector image*, in contrast, is a list of geometrical objects, such as lines, circles, arcs, or even cubes. These are much more difficult to produce, and so are mainly encountered as the output of sophisticated systems such as architectural and engineering CAD programs. Raster images are difficult to

modify because by definition they only encode single points: a line, for example, cannot grow or shrink as such, since it is not identified as such. Only its component parts are identified, and only they can be manipulated. Therefore the resolution or dot-size of a raster image is important, which is not the case with vector images. It is also far more difficult to convert raster images to vector images than to perform the opposite conversion. Raster images generally require more storage space than vector images, and a wide variety of methods exists for compressing them; the variation in these methods leads to corresponding variations in representations for storage and transmission of raster images.

Motion video usually consists of a long series of raster images. Data compression is even more effective on video than on single raster images (mainly owing to redundancy which arises from the usual similarity of adjacent frames). Notations for representing full-motion video are hotly debated at this time, and any user of these Guidelines would do well to obtain up-to-date expert advice before undertaking a project using them. The compression methods used with any of these image types may be "lossy" or "lossless". Methods for *lossy compression* save space by discarding a small portion of the image's detail, such as fine distinctions of shading. When decompressed, therefore, such an image will be only a close approximation of the original. In contrast, *lossless compression* guarantees that the exact uncompressed image will be reproducible from the compressed form: only truly redundant information is removed. In general, therefore, lossless compression does not save quite so much space as lossy compression, though it does guarantee fidelity to the original uncompressed image.

Raster images may be characterized by their *resolution*, which is the number of dots per inch used to represent the image. Doubling the resolution will give a more precise image, but also quadruple the storage requirement (before compression), and affect processing time for any operations to be performed, such as displaying an image for a reader. Motion video also has resolution in time: the number of frames to be shown per second. Encoders should consider carefully what resolution(s) and frame rate(s) to use for particular applications; these Guidelines express no recommendation in this matter, save the universal ones of consistency and documentation.

Within any image, it is typical to refer to locations via Cartesian co-ordinate axes: values for x, y, and sometimes z and/or time. These Guidelines provide for this via the SPACE keyword of the extended pointing mechanism discussed in section 14.2 ('Extended Pointers') on p. 340. However, graphic notations vary in whether co-ordinates count from left-to-right and top-to-bottom, or another way. They also vary in whether co-ordinates are considered real (inches, millimeters, and so on), or virtual (dots). These Guidelines do not recommend any of these methods over another, but all decisions made should be applied consistently, and documented in the <encodingDesc> section of the TEI header.[2] The way in which the color of an image is rendered also varies greatly. In monochrome images every displayed point is either black or white. In *gray-scale* images, each point is rendered in some shade of gray, the number of shades varying from system to system. In true polychrome images, points are rendered in different colours, again with varying limitations affecting the number of distinct shades and the means by which they are displayed.

22.5 Graphic Image Formats

As noted above, there exists a bewildering variety of different graphics formats, and the following list is in no way exhaustive. Moreover, inclusion of any format in this list should not be taken as indicating endorsement by the TEI of this format or any products associated with it. With the exception of CGM, all the formats listed here are proprietary to a greater or lesser extent and cannot therefore be regarded as standards in any meaningful sense. They are however widely used by many different vendors.

The following formats are widely used at the present time, and likely to remain supported by

[2] No special purpose element is provided for this purpose by the current version of the Guidelines. The information should be provided as one or more distinct paragraphs at the end of the <encDecl> element described in section 5.3 ('The Encoding Description') on p. 93.

more than one vendor's software:

- CGM: Computer Graphics Metafile
- PICT: Macintosh drawing format
- TIFF: Tagged Image File Format
- GIF: Graphics Interchange Format
- PBM: Portable Bit Map
- PCX: IBM PC raster format
- BMP: Microsoft bitmap format
- JPEG: Joint Photographic Expert Group
- QuickTime: Apple real-time image system
- Photo-CD: Kodak Photo Compact Disk format

Brief descriptions of all the above are given below. Where possible, current addresses or other contact information are shown for the originator of each format. Many formal standards, especially those promulgated by ISO and many related national organizations (ANSI, DIN, BSI, and many more), are available from those national organizations. Addresses may be found in any standard organizational directory for the country in question.

For each format, a sample notation declaration is given, using a formal public identifier constructed from the best information available at the date of publication. It is recommended that such formal public identifiers always be used in the interchange of documents between sites. Unless otherwise noted, however, these formal public identifiers have been formulated by the TEI, and not by the owners of the notation; if more recent versions of these formal public identifiers, or versions promulgated by the owners of the notation, are available at the time of document interchange, they should be used in preference to those shown here.

Support for formal public identifiers varies somewhat among existing SGML systems; for local processing, the notation declaration may therefore need to include a system identifier in addition to the formal public identifier. The documentation for the SGML system in use should be consulted for details.

22.5.1 Vector Graphic Formats

CGM: Computer Graphics Metafile This format is a popular representation for vector graphics specified by an ISO standard, ISO 8632:1987, amended in 1990. The standard defines binary, character, and plain-text encodings; the non-binary forms are safer for blind interchange, especially over networks. Documentation on CGM is available from ISO and from its member national bodies such as AFNOR, ANSI, BSI, DIN, JIS, etc. Sample declarations:

```
<!NOTATION cgm PUBLIC
    'ISO 8632:1987//NOTATION Computer Graphics Metafile//EN'>

<!NOTATION cgmchar  PUBLIC
    'ISO 8632-2:1987//NOTATION Character encoding//EN'>
<!NOTATION cgmbin   PUBLIC
    'ISO 8632-3:1987//NOTATION Binary encoding//EN'>
<!NOTATION cgmclear PUBLIC
    'ISO 8632-4:1987//NOTATION Clear text encoding//EN'>
```

PICT: Macintosh drawing format This format is universally supported on Macintosh(tm) systems, and readable by a limited range of software for other systems. Documentation is available from Apple Computer Company, Cupertino, California USA.

```
<!NOTATION pict PUBLIC
    '-//Apple Computer//NOTATION PICT:  Macintosh Drawing Format//EN'>
```

22.5.2 Raster Graphic Formats

TIFF: Tagged Image File Format Currently the most widely supported raster image format, especially for black and white images, TIFF is also one of the few formats commonly supported on more than one operating system. The drawback to TIFF is that it actually is a

wrapper for several formats, and some TIFF-supporting software does not support all variants. TIFF files may use LZW, CCITT Group 4, or PackBits compression methods, or may use no compression at all. Also, TIFF files may be monochrome, greyscale, or color. All such options should be specified in prose at the end of the <encodingDesc> section of the TEI header for any document including TIFF images. TIFF is owned by Aldus Corporation. Documentation on TIFF is available from them at Craigcook Castle, Craigcook Road, Edinburgh EH4 3UH, Scotland, or 411 First Avenue South, Seattle, Washington 98104 USA.

```
<!NOTATION tiff PUBLIC
   '-//Aldus Corporation//NOTATION Tagged Image File Format//EN'>
```

GIF: Graphics Interchange Format Color raster images are widely available in this form, which was created by CompuServ Information Services, but has by now been implemented for many other systems as well. Documentation on GIF is copyright by, and is available from, CompuServe Incorporated, Graphics Technology Department, 5000 Arlington Center Boulevard, Columbus, Ohio 43220 USA.

```
<!NOTATION gif PUBLIC
  '-//Compuserve Information Services//NOTATION
  Graphics Interchange Format//EN' >
```

PBM: Portable Bit Map PBM files are easy to process, eschewing all compression in favor of transparency of file format. PBM files can, of course, be compressed by generic file-compression tools for storage and transfer. Public domain software exists which will convert many other formats to and from PBM. Documentation on PBM is copyright by Jeff Poskanzer, and is available widely on the Internet.

```
<!NOTATION pbm PUBLIC
  '-//Jeff Poskanzer//NOTATION Portable Bit Map//EN' >
```

PCX: IBM PC raster format This format is used by most IBM PC paint programs, and supports both monochrome and color images. Documentation is available from ZSoft Corporation, Technical Support Department, ATTN: Technical Reference Manual, 450 Franklin Rd. Suite 100, Marietta, GA 30067 USA.

```
<!NOTATION pcx PUBLIC
  '-//ZSoft//NOTATION PCX:  IBM PC Raster Graphics Format//EN' >
```

BMP: Microsoft bitmap format This format is the standard raster format for computer using Microsoft Windows (tm) or Presentation Manager (tm). Documentation is available from Microsoft Corporation.

```
<!NOTATION bmp PUBLIC
  '-//MicroSoft//NOTATION BMP:  Bitmap Graphics Format//EN' >
```

22.5.3 Photographic and Motion Video Formats

JPEG: Joint Photographic Expert Group This standard is sponsored by CCITT and by ISO. It is ISO/IEC Draft International Standard 10918-1, and CCITT T.81. It handles monochrome and color images with a variety of compression techniques. JPEG per se, like CCITT Group IV, must be encapsulated before transmission; this can be done via TIFF, or via the JPEG File Interchange Format (JFIF).

```
<!NOTATION JPEG PUBLIC
  'ISO DIS 10918//NOTATION JPEG Graphics Format//EN' >
```

QuickTime: Apple real-time image system QuickTime is a proprietary method introduced by Apple Computer Company to synchronize the display of various data. The data can include frames of video, sound, lighting control equipment, and other things. Viewers for QuickTime productions are available for Apple and other computers. Further information is available from Apple Computer Incorporated, 10201 North de Anza Boulevard MS 23AQ, Cupertino, California 95014 USA.

```
<!NOTATION QuickTime PUBLIC
  '-//Apple Computer//NOTATION QuickTime Video Data Format//EN' >
```

Photo-CD: Kodak Photo Compact Disk format This format was introduced by Kodak for rasterizing photographs and storing them on CD-ROMs (about one hundred 35mm file images fit on one disk), for display on televisions or CD-I systems. Information on Photo-CD is available from Kodak Limited, Research and Development, Headstone Drive, Harrow, Middlesex HA1 4TY, UK.

```
<!NOTATION pcx PUBLIC
  '-//Eastman Kodak//NOTATION Photo-CD Raster Graphics Format//EN' >
```

As noted above, the reader will encounter many, many other graphics formats. Other formats are not recommended for data interchange according to the TEI scheme at this time, but may be included in a TEI document without affecting its conformance in other respects, provided that a notation declaration is provided.

Chapter 23

Language Corpora

The term *language corpus* is used to mean a number of rather different things. It may refer simply to any collection of linguistic data (written, spoken, or a mixture of the two), although many practitioners prefer to reserve it for collections which have been organized or collected with a particular end in view, generally to characterize a particular state or variety of one or more languages. Because opinions as to the best method of achieving this goal differ, various subcategories of corpora have also been identified. For our purposes however, the distinguishing characteristic of a corpus is that its components have been selected or structured according to some conscious set of design criteria.

These design criteria may be very simple and undemanding, or very sophisticated. A corpus may be intended to represent (in the statistical sense) a particular linguistic variety or sublanguage, or it may be intended to represent all aspects of some assumed "core" language. A corpus may be made up of whole texts or of fragments or text samples. It may be a "closed" corpus, or an "open" or "monitor" corpus, the composition of which may change over time. However, since an open corpus is of necessity finite at any particular point in time, the only likely effect of its expansibility from the encoding point of view may be some increased difficulty in maintaining consistent encoding practices (see further section 23.5 ('Recommendations for the Encoding of Large Corpora') on p. 555). For simplicity, therefore, our discussion largely concerns ways of encoding closed corpora, regarded as single but composite texts.

Language corpora are regarded by these Guidelines as *composite texts* rather than *unitary texts* (on this distinction, see chapter 7 ('Default Text Structure') on p. 183). This is because although each discrete sample of language in a corpus clearly has a claim to be considered as a text in its own right, it is also regarded as a subdivision of some larger object, if only for convenience of analysis. Corpora share a number of characteristics with other types of composite texts, including anthologies and collections. Most notably, different components of composite texts may exhibit different structural properties (for example, some may be composed of verse, and others of prose), thus potentially requiring elements from different TEI bases. Composite texts are thus especially likely to require the techniques for combining base tag sets described in section 3.4 ('Combining TEI Base Tag Sets') on p. 40.

Aside from these high-level structural differences, and possibly differences of scale, the encoding of language corpora and the encoding of individual texts present identical sets of problems. Any of the encoding techniques and elements presented in other chapters of these Guidelines may therefore prove relevant to some aspect of corpus encoding and may be used in corpora. However, we do not repeat here the discusssion of such fundamental matters as the representation of multiple character sets (see chapter 4 ('Characters and Character Sets') on p. 71); nor attempt to summarize the variety of elements provided for encoding basic structural features such as quoted or highlighted phrases, cross references, lists, notes, editorial changes and reference systems (see chapter 6 ('Elements Available in All TEI Documents') on p. 119). In addition to these general purpose elements, these Guidelines offer a range of more specialized sets of tags which may be of use in certain specialized corpora, for example those consisting primarily of verse (chapter 9 ('Base Tag Set for Verse ') on p. 213), drama (chapter 10 ('Base Tag Set for Drama') on p. 227), transcriptions of spoken text (chapter 11 ('Transcriptions of Speech')

on p. 249), etc. Chapter 3 ('Structure of the TEI Document Type Definition') on p. 35 should be reviewed for details of how these and other components of the Guidelines should be tailored to create a document type definition appropriate to a given application. In sum, it should not be asssumed that only the matters specifically addressed in this chapter are of importance for corpus creators.

This chapter does however include some other material relevant to corpora and corpus-building, for which no other location appeared suitable. It begins with a review of the distinction between unitary and composite texts, and of the different methods provided by these Guidelines for representing composite texts of different kinds (section 23.1 ('Varieties of Composite Text') on p. 538). Section 23.2 ('Contextual Information') on p. 540 describes a set of additional header elements provided for the documentation of contextual information, of importance largely though not exclusively to language corpora. This is the additional tag set for language corpora proper. Section 23.3 ('Associating Contextual Information with a Text') on p. 550 discusses a mechanism by which individual parts of the TEI Header may be associated with different parts of a TEI-conformant text. Section 23.4 ('Linguistic Annotation of Corpora') on p. 554 reviews various methods of providing linguistic annotation in corpora, with some specific examples of relevance to current practice in corpus linguistics. Finally, section 23.5 ('Recommendations for the Encoding of Large Corpora') on p. 555 provides some general recommendations about the use of these Guidelines in the building of large corpora.

23.1 Varieties of Composite Text

Both unitary and composite texts may be encoded using these Guidelines; composite texts, including corpora, will typically make use of the following tags for their top-level organization.

<teiCorpus.2> contains the whole of a TEI encoded corpus, comprising a single corpus header and one or more TEI.2 elements, each containing a single text header and a text.

<TEI.2> contains a single TEI-conformant document, comprising a TEI header and a text, either in isolation or as part of a <teiCorpus> element.

<teiHeader> supplies the descriptive and declarative information making up an "electronic title page" prefixed to every TEI-conformant text. Attributes include:

- **type** specifies the kind of document to which the header is attached. Legal values are:
  - ***text*** the header is attached to a single text.
  - ***corpus*** the header is attached to a corpus.
- **status** indicates whether the header is new or has been substantially revised. Sample values include:
  - ***new*** the header is a new header.
  - ***update*** the header is an update (has been revised).
- **creator** identifies the creator of the TEI Header.
- **date.created** indicates when the first version of the header was created.
- **date.updated** indicates when the current version of the header was created.

<text> contains a single text of any kind, whether unitary or composite, for example a poem or drama, a collection of essays, a novel, a dictionary, or a corpus sample.

<group> contains the body of a composite text, grouping together a sequence of distinct texts (or groups of such texts) which are regarded as a unit for some purpose, for example the collected works of an author, a sequence of prose essays, etc.

Full descriptions of these may be found in chapter 3 ('Structure of the TEI Document Type Definition') on p. 35 (for <TEI.corpus.2> and <TEI.2>), chapter 5 ('The TEI Header') on p. 77 (for <teiHeader>), and chapter 7 ('Default Text Structure') on p. 183 (for <text> and <group>); this section discusses their application to composite texts in particular.

In these Guidelines, the word *text* refers to any stretch of discourse, whether complete or incomplete, unitary or composite, which the encoder chooses (perhaps merely for purposes of analytic convenience) to regard as a unit. The term *composite text* refers to texts within which other texts appear; the following common cases may be distinguished:

- language corpora
- collections or anthologies
- poem cycles and epistolary works (novels or essays written in the form of collections or series of letters)
- otherwise unitary texts, within which one or more subordinate texts are embedded

The tags listed above may be combined to encode each of these varieties of composite text in different ways.

In corpora, the component samples are clearly distinct texts, but the systematic collection, standardized preparation, and common markup of the corpus often make it useful to treat the entire corpus as a unit, too. Some corpora may become so well established as to be regarded as texts in their own right; the Brown and LOB corpora are now close to achieving this status.

The <TEI.corpus.2> element is intended for the encoding of language corpora, though it may also be useful in encoding newspapers, electronic anthologies, and other disparate collections of material. The individual samples in the corpus are encoded as separate <TEI.2> elements, and the entire corpus is enclosed in a <TEI.corpus.2> element. Each sample has the usual structure for a <TEI.2> document, comprising a <teiHeader> followed by a <text> element. The corpus, too, has a corpus-level <teiHeader> element, in which the corpus as a whole, and encoding practices common to multiple samples may be described. The overall structure of a TEI-conformant corpus is thus:

```
<TEI.corpus.2>
   <teiHeader type=corpus> ...
   <!-- TEI header for corpus-level information -->
   </teiHeader>

   <TEI.2 id=T1>
        <teiHeader type=text> ... </teiHeader>
        <text> ... </text>
   </TEI.2>
   <TEI.2 id=T2>
        <teiHeader type=text> ... </teiHeader>
        <text> ... </text>
   </TEI.2>
   <!-- ... etc. -->
</TEI.corpus.2>
```

Header information which relates to the whole corpus rather than to individual components of it should be factored out and included in the <teiHeader> element prefixed to the whole. This two-level structure allows for contextual information to be specified at the corpus level, at the individual text level, or at both. Discussion of the kinds of information which may thus be specified is provided below, in section 23.2 ('Contextual Information') on p. 540, as well as in chapter 5 ('The TEI Header') on p. 77. Information of this type should in general be specified only once: a variety of methods are provided for associating it with individual components of a corpus, as further described in section 23.3 ('Associating Contextual Information with a Text') on p. 550.

In some cases, the design of a corpus is reflected in its internal structure. For example, a corpus of newspaper extracts might be arranged to combine all stories of one type (reportage, editorial, reviews, etc.) into some higher-level grouping, possibly with sub-groups for date, region, etc. The <TEI.corpus.2> element provides no direct support for reflecting such internal corpus structure in the markup: it treats the corpus as an undifferentiated series of components, each tagged <TEI.2>.

If it is essential to reflect a single permanent organization of a corpus into sub- and sub-sub-corpora, then the corpus or the high-level subcorpora may be encoded as composite texts, using the <group> element described below and in section 7.3 ('Groups of Texts') on p. 195.

The mechanisms for corpus characterization described in this chapter, however, are designed to reduce the need to do this. Useful groupings of components may easily be expressed using the text classification and identification elements described in section 23.2.1 ('The Text Description') on p. 541, and those for associating declarations with corpus components described in section 23.3 ('Associating Contextual Information with a Text') on p. 550. These methods also allow several different methods of text grouping to co-exist, each to be used as needed at different times. This helps minimize the danger of cross-classification and mis-classification of samples, and helps improve the flexibility with which parts of a corpus may be characterized for different applications.

Anthologies and collections are often treated as texts in their own right, if only for historical reasons. In conventional publishing, at least, anthologies are published as units, with single editorial responsibility and common front and back matter which may need to be included in their electronic encodings. The texts collected in the anthology, of course, may also need to identifiable as distinct individual objects for study.

Poem cycles, epistolary novels, and epistolary essays differ from anthologies in that they are often written as single works, by single authors, for single occasions; nevertheless, it can be useful to treat their constituent parts as individual texts, as well as the cycle itself. Structurally, therefore, they may be treated in the same way as anthologies: in both cases, the body of the text is composed largely of other texts.

The <group> element is provided to simplify the encoding of collections, anthologies, and cyclic works; as noted above, the <group> element can also be used to record the potentially complex internal structure of language corpora. For full description, see chapter 7 ('Default Text Structure') on p. 183.

Some composite texts, finally, are neither corpora, nor anthologies, nor cyclic works: they are otherwise unitary texts within which other texts are embedded. In general, they may be treated in the same way as unitary texts, using the normal <TEI.2> and <body> elements. The embedded text itself may be encoded using the <text> element, which may occur within quotations or between paragraphs or other chunk-level elements inside the sections of a larger text. For further discussion, see chapter 7 ('Default Text Structure') on p. 183.

All composite texts share the characteristic that their different component texts may be of structurally similar or dissimilar types. If all component texts may all be encoded using the same base tag set, then no problem arises. If however they require different base tag sets, then either the general or the mixed base tag set must be used, in addition to all relevant base tag sets. This process is described in more detail in section 3.4 ('Combining TEI Base Tag Sets') on p. 40.

23.2 Contextual Information

Contextual information is of particular importance for collections or corpora composed of samples from a variety of different kinds of text. Examples of such contextual information include: the age, sex and geographical origins of participants in a language interaction, or their socio-economic status; the cost and publication data of a newspaper; the topic, register or factuality of an extract from a textbook. Such information may be of the first importance, whether as an organizing principle in creating a corpus (for example, to ensure that the range of values in such a parameter is evenly represented throughout the corpus, or represented proportionately to the population being sampled), or as a selection criterion in analysing the corpus (for example, to investigate the language usage of some particular vector of social characteristics).

Such contextual information is potentially of equal importance for unitary texts, and these Guidelines accordingly make no particular distinction between the kinds of information which should be gathered for unitary and for composite texts. In either case, the information should be recorded in the appropriate section of a TEI Header, as described in chapter 5 ('The TEI Header') on p. 77. In the case of language corpora, such information may be gathered together in the overall corpus header, or split across all the component texts of a corpus, in their individual headers, or divided between the two. The association between an individual corpus text and the contextual information applicable to it may be made in a number of ways, as further discussed in

section 23.3 ('Associating Contextual Information with a Text') on p. 550 below.

Chapter 5 ('The TEI Header') on p. 77, which should be read in conjunction with the present section, describes in full the range of elements available for the encoding of information relating to the electronic file itself, for example its bibliographic description and those of the source or sources from which it was derived (see section 5.2 ('The File Description') on p. 80); information about the encoding practices followed with the corpus, for example its design principles, editorial practices, reference system etc. (see section 5.3 ('The Encoding Description') on p. 93); more detailed descriptive information about the creation and content of the corpus, such as the languages used within it and any descriptive classification system used (see section 5.4 ('The Profile Description') on p. 108); and version information documenting any changes made in the electronic text (see section 5.5 ('The Revision Description') on p. 112).

In addition to the elements defined by chapter 5 ('The TEI Header') on p. 77, several other elements can be used in the TEI header if the additional tag set defined by this chapter is invoked. These additional tags make it possible to characterize the social or other situation within which a language interaction takes place or is experienced, the physical setting of a language interaction, and the participants in it. Though this information may be relevant to, and provided for, unitary texts as well as for collections or corpora, it is more often recorded for the components of systematically developed corpora than for isolated texts, and thus the additional tag set is referred to as being "for language corpora". Included in this tag set are the following elements:

<textDesc> provides a description of a text in terms of its *situational parameters*.

<particDesc> describes the identifiable speakers, voices or other participants in a linguistic interaction.

<settingDesc> describes the setting or settings within which a language interaction takes place, either as a prose description or as a series of <setting> elements.

These elements form an optional extension to the <profileDesc>, defined in section 5.4 ('The Profile Description') on p. 108 and are further described in the remainder of this section. They are formally defined as follows:

```
<!-- 23.2:  Header extensions for Corpus Texts                -->
<!ELEMENT textDesc      - o  (channel, constitution,
                              derivation, domain, factuality,
                              interaction, preparedness,
                              purpose+)                         >
<!ATTLIST textDesc           %a.global;
                              %a.declarable;                    >
<!ELEMENT particDesc    - o  (p+ | ( (person | personGrp)+,
                              particLinks?) )                   >
<!ATTLIST particDesc         %a.global;
                              %a.declarable;                    >
<!ELEMENT settingDesc   - o  (p+ | setting+)                    >
<!ATTLIST settingDesc        %a.global;
                              %a.declarable;                    >
<!-- (continued in sec. 23.2.1, 23.2.2, 23.2.3)                -->
```

The additional tag set for language corpora will be invoked, thus enabling the use of these elements, if a parameter entity called *TEI.corpus* is declared with the value INCLUDE, somewhere within the DTD subset. If the document is structured as a TEI corpus (that is, using the <TEI.corpus.2> element), its document type declaration will resemble this:

```
<!DOCTYPE TEI.corpus.2 system 'TEI2.dtd' [
    <!ENTITY % TEI.corpus 'INCLUDE' >
]>
```

23.2.1 The Text Description

The <textDesc> element provides a full description of the situation within which a text was produced or experienced, and thus characterizes it in a way relatively independent of any *a priori* theory of text-types. It is provided as an alternative or a supplement to the common use of

descriptive taxonomies used to categorize texts, which is fully described in section 5.4.3 ('The Text Classification') on p. 111, and section 5.3.6 ('The Classification Declaration') on p. 104. The description is organized as a set of values and optional prose descriptions for the following eight *situational parameters*, each represented by one of the following eight elements:

<channel> describes the medium or channel by which a text is delivered or experienced. For a written text, this might be print, manuscript, e-mail, etc.; for a spoken one, radio, telephone, face-to-face, etc. Attributes include:

mode specifies the mode of this channel with respect to speech and writing. Legal values are:

s spoken
w written
sw spoken to be written (e.g. dictation)
ws written to be spoken (e.g. a script)
m mixed modes
x unknown or inapplicable

<constitution> describes the internal composition of a text or text sample, for example as fragmentary, complete, etc. Attributes include:

type specifies how the text was constituted. Legal values are:

single a single complete text
composite a text made by combining several smaller items, each individually complete
frags a text made by combining several smaller, not necessarily complete, items
unknown composition unknown or unspecified

<derivation> describes the nature and extent of indebtedness or derivativeness of this text with respect to others. Attributes include:

type categorizes the derivation of the text. Sample values include:

original text is original
revision text is a revision of some other text
translation text is a translation of some other text
abridgment text is an abridged version of some other text
plagiarism text is plagiarized from some other text
traditional text has no obvious source but is one of a number derived from some common ancestor

<domain> describes the most important social context in which the text was realized or for which it is intended, for example private vs. public, education, religion, etc. Attributes include:

type categorizes the domain of use. Sample values include:

art art and entertainment
domestic domestic and private
religious religious and ceremonial
business business and work place
education education
govt government and law
public other forms of public context

<factuality> describes the extent to which the text may be regarded as imaginative or non-imaginative, that is, as describing a fictional or a non-fictional world. Attributes include:

type categorizes the factuality of the text. Legal values are:

fiction the text is to be regarded as entirely imaginative
fact the text is to be regarded as entirely informative or factual
mixed the text contains a mixture of fact and fiction
inapplicable the fiction/fact distinction is not regarded as helpful or appropriate to this text

<interaction> describes the extent, cardinality and nature of any interaction among those producing and experiencing the text, for example in the form of response or interjection, commentary etc. Attributes include:

type specifies whether or not there is any interaction between active and passive participants in the text. Legal values are:
- ***none*** no interaction of any kind, e.g. a monologue
- ***partial*** some degree of interaction, e.g. a monologue with set responses
- ***complete*** complete interaction, e.g. a face to face conversation
- ***inapplicable*** this parameter is inappropriate or inapplicable in this case

active specifies the number of active participants (or *addressors*) producing parts of the text. Legal values are:
- ***singular*** a single addressor
- ***plural*** many addressors
- ***corporate*** a corporate addressor
- ***unknown*** number of addressors unknown or unspecifiable

passive specifies the number of passive participants (or *addressees*) to whom a text is directed or in whose presence it is created or performed. Suggested values include:
- ***self*** text is addressed to the originator e.g. a diary
- ***single*** text is addressed to one other person e.g. a personal letter
- ***many*** text is addressed to a countable number of others e.g. a conversation in which all participants are identified
- ***group*** text is addressed to an undefined but fixed number of participants e.g. a lecture
- ***world*** text is addressed to an undefined and indeterminately large number e.g. a published book

<preparedness> describes the extent to which a text may be regarded as prepared or spontaneous. Attributes include:

type a keyword characterizing the type of preparedness. Sample values include:
- ***none*** spontaneous or unprepared
- ***scripted*** follows a script
- ***formulaic*** follows a predefined set of conventions
- ***revised*** polished or revised before presentation

<purpose> characterizes a single purpose or communicative function of the text. Attributes include:

type specifies a particular kind of purpose. Suggested values include:
- ***persuade*** didactic, advertising, propaganda, etc.
- ***express*** self expression, confessional, etc.
- ***inform*** convey information, educate, etc.
- ***entertain*** amuse, entertain, etc.

degree specifies the extent to which this purpose predominates. Legal values are:
- ***high*** this purpose is predominant
- ***medium*** this purpose is intermediate
- ***low*** this purpose is weak
- ***unknown*** extent unknown

A TEI-conformant text description contains each of the above elements, supplied in the order specified. Except for the <purpose> element, which may be repeated to indicate multiple purposes, no element may appear more than once within a single text description. Each element may be empty, or may contain a brief qualification or more detailed description of the value expressed by its attributes. It should be noted that some texts, in particular literary ones, may resist unambiguous classification in some of these dimensions; in such cases, the situational parameter in question should be given the content "not applicable" or an equivalent phrase.

Texts may be described along many dimensions, according to many different taxonomies. No generally accepted consensus as to how such taxonomies should be defined has yet emerged, despite the best efforts of many corpus linguists, text linguists, sociolinguists, rhetoricians, and literary theorists over the years. Rather than attempting the task of proposing a single taxonomy of *text-types* (or the equally impossible one of enumerating all those which have been proposed previously), the closed set of *situational parameters* described above can be used in combination to

supply useful distinguishing descriptive features of individual texts, without insisting on a system of discrete high-level text-types. Such text-types may however be used in combination with the parameters proposed here, with the advantage that the internal structure of each such text-type can be specified in terms of the parameters proposed. This approach has the following analytical advantages:[1]

- it enables a relatively continuous characterization of texts (in contrast to discrete categories based on type or topic)
- it enables meaningful comparisons across corpora
- it allows analysts to build and compare their own text-types based on the particular parameters of interest to them
- it is equally applicable to spoken and written texts

Two alternative approaches to the use of these parameters are supported by these Guidelines. One is to use pre-existing taxonomies such as those used in subject classification or other types of text categorization. Such taxonomies may also be appropriate for the description of the topics addressed by particular texts. Elements for this purpose are described in section 5.4.3 ('The Text Classification') on p. 111, and elements for defining or declaring such classification schemes in section 5.3.6 ('The Classification Declaration') on p. 104. A second approach is to develop an application-specific set of *feature structures* and an associated *feature system declaration*, as described in chapters 16 ('Feature Structures') on p. 397 and 26 ('Feature System Declaration') on p. 589.

Where the organizing principles of a corpus or collection so permit, it may be convenient to regard a particular set of values for the situational parameters listed in this section as forming a *text-type* in its own right; this may also be useful where the same set of values applies to several texts within a corpus. In such a case, the set of text-types so defined should be regarded as a *taxonomy*. The mechanisms described in section 5.3.6 ('The Classification Declaration') on p. 104 may be used to define hierarchic taxonomies of such text-types, provided that the <catDesc> component of the <category> element contains a <textDesc> element rather than a prose description. Particular texts may then be associated with such definitions using the mechanisms described in sections 5.4.3 ('The Text Classification') on p. 111.

Using these situational parameters, an informal domestic conversation might be characterized as follows:

```
<textDesc id=t1 n='Informal domestic conversation'>
  <channel mode=s>informal face-to-face conversation</channel>
  <constitution type=single>each text represents a continuously
        recorded interaction among the specified participants
        </constitution>
  <derivation type=original>
  <domain type=domestic>plans for coming week, local affairs</domain>
  <factuality type=mixed>mostly factual, some jokes</factuality>
  <interaction type=complete active=plural passive=many>
  <preparedness type='spontaneous'>
  <purpose type=entertain degree=high>
  <purpose type=inform degree=medium>
 </textDesc>
```

The following example demonstrates how the same situational parameters might be used to characterize a novel:

```
<textDesc n='novel'>
  <channel mode=w>print; part issues</channel>
  <constitution type=single>
  <derivation type=original>
  <domain type=art>
  <factuality type=fiction>
  <interaction type=none>
```

[1] Schemes similar to that proposed here were developed in the 1960s and 1970s by researchers such as Hymes, Halliday, and Crystal and Davy, but have rarely been implemented; one notable exception being the pioneering work on the Helsinki Diachronic Corpus of English, on which see M. Kytö and M. Rissanen, *The Helsinki Corpus of English Texts*, in *Corpus Linguistics: hard and soft*, ed. M. Kytö, O. Ihalainen, and M. Rissanen (Amsterdam: Rodopi, 1988).

```
    <preparedness type=prepared>
    <purpose type=entertain degree=high>
    <purpose type=inform degree=medium>
  </textDesc>
```

The formal declarations for these elements are given below:

```
<!-- 23.2.1:  Header extensions for Corpus Texts (cont'd)     -->
<!-- (continuation of sec. 23.2)                              -->
<!ELEMENT channel       - o  (%phrase.seq)                     >
<!ATTLIST channel            %a.global;
          mode               (s | w | ws | sw | m | x)
                                                  x           >
<!ELEMENT constitution  - o  (%phrase.seq)                     >
<!ATTLIST constitution       %a.global;
          type               (single | composite | frags |
                             unknown)             single       >
<!ELEMENT derivation    - o  (%phrase.seq)                     >
<!ATTLIST derivation         %a.global;
          type               CDATA                #IMPLIED     >
<!ELEMENT domain        - o  (%phrase.seq)                     >
<!ATTLIST domain             %a.global;
          type               CDATA                #IMPLIED     >
<!ELEMENT factuality    - o  (%phrase.seq)                     >
<!ATTLIST factuality         %a.global;
          type               (fiction | fact | mixed |
                             inapplicable)        #IMPLIED     >
<!ELEMENT interaction   - o  (%phrase.seq;)                    >
<!ATTLIST interaction        %a.global;
          type               (none | partial | complete |
                             inapplicable)        #IMPLIED
          active             CDATA                #IMPLIED
          passive            CDATA                #IMPLIED     >
<!ELEMENT preparedness  - o  (%phrase.seq)                     >
<!ATTLIST preparedness       %a.global;
          type               CDATA                #IMPLIED     >
<!ELEMENT purpose       - o  (%phrase.seq)                     >
<!ATTLIST purpose            %a.global;
          type               CDATA                #IMPLIED
          degree             (high | medium | low | unknown)
                                                  #IMPLIED     >
```

23.2.2 The Participants Description

The <particDesc> element in the <profileDesc> element provides additional information about the participants in a spoken text or, where this is judged appropriate, the persons named or depicted in a written text. Individual speakers or groups of speakers may be named or identified by a code which can then be used elsewhere within the encoded text, for example as the value of a who attribute. Demographic and descriptive information may be supplied about their individual characteristics and the relationships between them.

It should be noted that although the terms *speaker* or *participant* are used throughout this section, it is intended that the same mechanisms may be used to characterize fictional personæ or "voices" within a written text, except where otherwise stated. For the purposes of analysis of language usage, the information specified here should be equally applicable to written and spoken texts.

The element <particDesc> contains one or more <person> or <personGrp> elements, followed by an optional <particLinks> element, as described below:

<person> describes a single participant in a language interaction. Attributes include:

role specifies the role of this participant in the group.

sex specifies the sex of the participant. Sample values include:

m male
f female
u unknown or inapplicable

age specifies the age group to which the participant belongs.

<personGrp> describes a group of individuals treated as a single person for analytic purposes. Attributes include:

role specifies the role of this group of participants in the interaction.
sex specifies the sex of the participant group. Sample values include:
m male
f female
u unknown
x mixed
age specifies the age group of the participants.
size specifies the size or approximate size of the group.

<particLinks> describes the relationships or social links existing between participants in a linguistic interaction.

Both <person> and <personGrp> elements have the same substructure. This may be a prose description, or, more formally, a series of specialized subelements providing more specific details. Such details will vary enormously for different kinds of analysis; the set of demographic characteristics presented here as sub-elements should therefore be regarded as providing only an indication of the kinds of descriptive information which have been found to be generally useful, for example in socio-linguistics. Users of these Guidelines are free to extend or modify this set of demographic characteristics, by redefining the parameter entity *m.demographics*, associated with the class *demographics*, as further described in chapter 29 ('Modifying the TEI DTD') on p. 619. Where well-known classification schemes exist, e.g. for socio-economic class or occupation, these should be used and may be documented in the same way as for text classification (see section 5.3.6 ('The Classification Declaration') on p. 104)

The following elements are the default members of the class *demographics*:

<birth> contains information about a person's birth, such as its date and place. Attributes include:

date specifies the date of birth in an ISO standard form (yyyy-mm-dd).

<firstLang> specifies the first language of a participant.

<langKnown> contains an informal description of a person's competence in different languages, dialects, etc.

<residence> describes a person's present or past places of residence.

<education> contains a brief prose description of the educational background of a participant.

<affiliation> contains an informal description of a person's present or past affiliation with some organization, for example an employer or sponsor.

<occupation> contains an informal description of a person's trade, profession or occupation. Attributes include:

scheme identifies the classification system or taxonomy in use by supplying the identifier of a <taxonomy> element elsewhere in the header.
code identifies an occupation code defined within the classification system or taxonomy defined by the source attribute.

<socecStatus> contains an informal description of a person's perceived social or economic status. Attributes include:

scheme identifies the classification system or taxonomy in use.
code identifies a status code defined within the classification system or taxonomy defined by the source attribute.

For example, an individual might be described informally by the following <person> element:

```
<person id=P1 sex=F age='mid'>
  <p>Female informant, well-educated, born in Shropshire
```

```
    UK, 12 Jan 1950, of unknown occupation.
    Speaks French fluently. Socio-Economic status B2
    in the PEP classification scheme.
</person>
```

Provided that the "PEP classification scheme" has been defined elsewhere in the heading (as a <taxonomy> element within the <textClass> element; see 5.3.6 ('The Classification Declaration') on p. 104), the same individual might more formally be described as follows:

```
<person id=P1 sex=F age='mid'>
  <birth date='1950-01-12'>
     <date>12 Jan 1950</date>
     <name type=place>Shropshire, UK</name>
  </birth>
  <firstLang>English</firstLang>
  <langKnown>French</langKnown>
  <residence>Long term resident of Hull</residence>
  <education>University postgraduate</education>
  <occupation>Unknown</occupation>
  <socecstatus source=PEP code=B2>
</person>
```

Dates and names of persons or places, if included in the prose description, may be encoded using either the general purpose <date>, <name> or <rs> elements discussed in section 6.4 ('Names, Numbers, Dates, Abbreviations, and Addresses') on p. 132, or the more specialised and detailed elements provided by chapter 20 ('Names and Dates') on p. 487. In the latter case, the additional tag set for names and dates must be enabled together with that for language corpora.

An identified character in a drama or a novel might be defined using a subset of the same tags as follows:[2]

```
<person id=EM01 sex=F age=young>
<p><name>Emma Woodhouse</name>
</person>
```

As noted above, the <particLinks> element is used to document personal or social relationships between individual participants, where this is felt to be of importance in the analysis. This may be done either as an informal prose description, or more formally using the special purpose <relation> element, as described below:

<relation> describes any kind of relationship or linkage amongst a specified group of participants. Attributes include:

- **type** categorizes the relationship in some respect, e.g. as social, personal or other. Suggested values include:
  - ***social*** relationship concerned with social roles
  - ***personal*** relationship concerned with personal roles, e.g. kinship, marriage, etc.
  - ***other*** other kinds of relationship
- **desc** briefly describes the relationship.
- **active** identifies the "active"participants in a non-mutual relationship, or all the participants in a mutual one.
- **passive** identifies the "passive" participants in a non-mutual relationship.
- **mutual** indicates whether the relationship holds equally amongst all the participants. Legal values are:
  - ***Y*** the relationship is mutual
  - ***N*** the relationship is directed

A *relationship*, as defined here, may be any kind of describable link between specified participants, for example a social relationship (such as employer/employee), a personal relationship (such as sibling, spouse, etc.) or something less precise such as "possessing shared knowledge". A relationship may be *mutual*, in that all the participants engage in it on an equal footing (for

[2] It is particularly useful to define participants in a dramatic text in this way, since it enables the who attribute to be used to link <sp> elements to definitions for their speakers; see further section 10.2.2 ('Speeches and Speakers') on p. 237.

example the "sibling" relationship); or it may not be if participants are not identical with respect to their role in the relationship (for example, the "employer" relationship). For non-mutual relationships, only two kinds of role are currently supported; they are named *active* and *passive*. These names are chosen to reflect the fact that non-mutual relations are *directed*, in the sense that they are most readily described by a transitive verb, or a verb phrase of the form 'is X of' or 'is X to'. The subject of the verb is classed as *active*; the direct object of the verb, or the object of the concluding preposition, as *passive*. Thus parents are "active" and children "passive" in the relationship "parent" (interpreted as 'is parent of'); the employer is "active", the employee "passive", in the relationship 'employs'. These relationships can be inverted: parents are "passive" and children "active" in the relationship 'is child of'; similarly "works for" inverts the active and passive roles of "employs".

For example:

```
<particLinks>
  <relation desc='parent' active='P1 P2'
         passive='P3 P4' mutual=N>
  <relation desc='spouse' active='P1 P2'>
  <relation class='social' desc='employer'
          active=P1 passive='P3 P5 P6 P7' mutual=N>
</particLinks>
```

This example defines the following three relationships among participants P1 through P7:

P1 and P2 are parents of P3 and P4.
P1 and P2 are linked in a mutual relationship called "spouse" — i.e. P2 is the spouse of P1, and P1 is the spouse of P2.
P1 has the social relationship "employer" with respect to P3, P5, P6, and P7.

The elements discussed in this section are formally defined as follows:

```
<!-- 23.2.2:  Header extensions for Corpus Texts (cont'd)     -->
<!-- (continuation of sec. 23.2)                              -->
<!ENTITY % x.demographic ''                                    >
<!ENTITY % m.demographic '%x.demographic birth | education |
           firstLang | occupation | persName | residence |
           socecStatus'                                        >
<!ELEMENT person        - o  (p+ | (%m.demographic)* )         >
<!ATTLIST person             %a.global;
          role               CDATA              #IMPLIED
          sex                (m | f | u )       #IMPLIED
          age                CDATA              #IMPLIED      >
<!ELEMENT personGrp     - o  (p+ | (%m.demographic)* )         >
<!ATTLIST personGrp          %a.global;
          role               CDATA              #IMPLIED
          sex                (m | f | u | x)    #IMPLIED
          age                CDATA              #IMPLIED
          size               CDATA              #IMPLIED      >
<!ELEMENT birth         - -  (%phrase.seq)                     >
<!ATTLIST birth              %a.global;
          date               %ISO-date          #IMPLIED      >
<!ELEMENT firstLang     - o  (%phrase.seq)                     >
<!ATTLIST firstLang          %a.global;                        >
<!ELEMENT langKnown     - -  (%phrase.seq)                     >
<!ATTLIST langKnown          %a.global;                        >
<!ELEMENT residence     - o  (%phrase.seq)                     >
<!ATTLIST residence          %a.global;                        >
<!ELEMENT education     - o  (%phrase.seq)                     >
<!ATTLIST education          %a.global;                        >
<!ELEMENT affiliation   - -  (%phrase.seq)                     >
<!ATTLIST affiliation        %a.global;                        >
<!ELEMENT occupation    - -  (%phrase.seq)                     >
<!ATTLIST occupation         %a.global;
          scheme             IDREF              #IMPLIED
```

```
          code              IDREF              #IMPLIED       >
<!ELEMENT socecStatus   - o (%phrase.seq)                     >
<!ATTLIST socecStatus       %a.global;
          scheme            IDREF              #IMPLIED
          code              IDREF              #IMPLIED       >
<!ELEMENT particLinks   - o (p+ | relation+)                  >
<!ATTLIST particLinks       %a.global;                        >
<!ELEMENT relation      - o EMPTY                             >
<!ATTLIST relation          %a.global;
          type              CDATA              personal
          desc              CDATA              #IMPLIED
          active            IDREFS             #IMPLIED
          passive           IDREFS             #IMPLIED
          mutual            (Y | N)            Y              >
```

23.2.3 The Setting Description

The <settingDesc> element is used to describe the setting or settings in which language interaction takes place. It may contain a prose description, analogous to a stage description at the start of a play, stating in broad terms the locale, or a more detailed description of a series of such settings. Individual settings may be associated with particular participants by means of the optional who attribute if, for example, participants are in different places. This attribute identifies one or more individual participants or participant groups, as discussed earlier in section 23.2.2 ('The Participants Description') on p. 545. If this attribute is not specified, the setting details provided are assumed to apply to all participants represented in the language interaction. [3]

Each distinct setting is described by means of a <setting> element, which contains either a prose description or a combination of the other elements listed below:

<setting> describes one particular setting in which a language interaction takes place. Attributes include:

who supplies the identifiers of the participants at this setting.

<name> contains a proper noun or noun phrase. Attributes include:

type indicates the type of the object which is being named by the phrase.

<date> contains a date in any format. Attributes include:

value gives the value of the date in some standard form, usually yyyy-mm-dd.

<time> contains a phrase defining a time of day in any format. Attributes include:

value gives the value of the time in a standard form.

<locale> contains a brief informal description of the nature of a place for example a room, a restaurant, a park bench etc.

<activity> contains a brief informal description of what a participant in a language interaction is doing other than speaking, if anything.

The following example demonstrates the kind of background information often required to support transcriptions of language interactions, first encoded as a simple prose narrative:

```
<settingDesc>
<p>The time is early spring, 1989.  P1 and P2 are
 playing on the rug of a suburban home in Bedford.
 P3 is doing the washing up at the sink.  P4 (a
 radio announcer) is in a broadcasting studio in London.
</settingDesc>
```

The same information might be represented more formally in the following way:

```
<settingDesc>
<setting who="P1 P2">
 <name type=city>Bedford</>
```

[3] The present proposals do not support the encoding of different settings for the same participant. This is a subject for further work.

```
  <name type=region>UK: South East</>
  <date value=1989>early spring, 1989</>
  <locale>rug of a suburban home</>
  <activity>playing</>
 </setting>
 <setting who=P3>
  <name type=city>Bedford</>
  <name type=region>UK: South East</>
  <date value=1989>early spring, 1989</>
  <locale>at the sink</>
  <activity>washing-up</>
 </setting>
 <setting who=P4>
  <name type=place>London, UK</>
  <time>unknown</>
  <locale>broadcasting studio</>
  <activity>radio performance</>
 </setting>
</settingDesc>
```

For more detailed encoding of names of persons and places, the additional tag set described in chapter 20 ('Names and Dates') on p. 487 may additionally be used; the above examples assume that only the general purpose <name> element supplied in the core tag set is available. The elements discussed in this section have the following formal definitions:

```
<!-- 23.2.3:  Header extensions for Corpus Texts (cont'd)     -->
<!-- (continuation of sec. 23.2)                              -->
<!ELEMENT setting       - -  (p+ | (name | time | locale |
                              activity)* )                     >
<!ATTLIST setting             %a.global;
          who                 IDREFS              #IMPLIED     >
<!ELEMENT locale        - o  (%phrase.seq)                      >
<!ATTLIST locale              %a.global;                        >
<!ELEMENT activity      - o  (%phrase.seq)                      >
<!ATTLIST activity            %a.global;                        >
```

23.3 Associating Contextual Information with a Text

This section discusses the assocation of the contextual information held in the header with the individual elements making up a TEI text or corpus. Contextual information is held in elements of various kinds within the TEI header, as discussed elsewhere in this section and in chapter 5 ('The TEI Header') on p. 77. Here we consider what happens when different parts of a document need to be associated with different contextual information of the same type, for example when one part of a document uses a different encoding practice from another, or where one part relates to a different setting from another. In such situations, there will be more than one instance of a header element of the relevant type.

The TEI DTDs allow for the following possibilities:

- A given element may appear in the corpus header only, in the header of one or more texts only, or in both places
- There may be multiple occurrences of certain elements in either corpus or text header.

To simplify the exposition, we deal with these two possibilities separately in what follows; however, they may be combined as desired.

23.3.1 Combining Corpus and Text Headers

A TEI conformant document may have more than one header only in the case of a TEI corpus, which must have a header in its own right, as well as the obligatory header for each text. Every

element specified in a corpus-header is understood as if it appeared within every text header in the corpus. An element specified in a text header but not in the corpus header supplements the specification for that text alone. If any element is specified in both corpus and text headers, the corpus header element is over-ridden for that text alone.

The <titleStmt> for a corpus text is understood to be prefixed by the <titleStmt> given in the corpus header. All other optional elements of the <fileDesc> should be omitted from an individual corpus text header unless they differ from those specified in the corpus header. All other header elements behave identically, in the manner documented below. This facility makes it possible to state once for all in the corpus header each piece of contextual information which is common to the whole of the corpus, while still allowing for individual texts to vary from this common denominator.

For example, the following schematic shows the structure of a corpus comprising three texts, the first and last of which share the same encoding declaration. The second one has its own encoding declaration

```
<TEI.corpus.2>
<TeiHeader>
    <!-- contains declarations common to the whole corpus -->
    <fileDesc> ... </>
    <encodingDesc> ...
        <!-- for example, this encodingDesc is common to
             whole corpus. --> ... </>
    <revisionDesc> ... </>
</TeiHeader>

<TEI.2>
<TeiHeader>
   <fileDesc>
        <!-- details peculiar to this text --> ... </>
</TeiHeader>
<text>
 ...
</TEI.2>

<TEI.2>
<TeiHeader>
   <fileDesc>
        <!-- details peculiar to this text --> ... </>
   <encodingDesc>
        <!-- encoding description
             peculiar to this text --> ... </>
</TeiHeader>
<text>
 ...
</TEI.2>

<TEI.2>
<TeiHeader>
   <fileDesc><!-- details peculiar to this text --> ... </>
</TeiHeader>
<text>
 ...
</TEI.2>
</TEI.corpus.2>
```

23.3.2 Declarable Elements

Certain of the elements which can appear within a TEI Header are known as *declarable elements*. These elements have in common the fact that they may be linked explicitly with a particular part of a text or corpus by means of a **decls** attribute. This linkage is used to over-ride the default association between declarations in the header and a corpus or corpus text. The only header elements which may be associated in this way are those which would not otherwise be meaningfully repeatable.

An alphabetically ordered list of declarable elements follows:

<bibl> contains a loosely-structured bibliographic citation of which the sub-components may or may not be explicitly tagged.

<biblFull> contains a fully-structured bibliographic citation, in which all components of the TEI file description are present.

<biblStruct> contains a structured bibliographic citation, in which only bibliographic subelements appear and in a specified order.

<broadcast> describes a broadcast used as the source of a spoken text.

<correction> states how and under what circumstances corrections have been made in the text.

<editorialDecl> provides details of editorial principles and practices applied during the encoding of a text.

<equipment> provides technical details of the equipment and media used for an audio or video recording used as the source for a spoken text.

<hyphenation> summarizes the way in which hyphenation in a source text has been treated in an encoded version of it.

<interpretation> describes the scope of any analytic or interpretive information added to the text in addition to the transcription.

<langUsage> describes the languages, sublanguages, registers, dialects etc. represented within a text.

<listBibl> contains a list of bibliographic citations of any kind.

<normalization> indicates the extent of normalization or regularization of the original source carried out in converting it to electronic form.

<particDesc> describes the identifiable speakers, voices or other participants in a linguistic interaction.

<projectDesc> describes in detail the aim or purpose for which an electronic file was encoded, together with any other relevant information concerning the process by which it was assembled or collected.

<quotation> specifies editorial practice adopted with respect to quotation marks in the original.

<recording> details of an audio or video recording event used as the source of a spoken text, either directly or from a public broadcast.

<samplingDecl> contains a prose description of the rationale and methods used in sampling texts in the creation of a corpus or collection.

<scriptStmt> contains a citation giving details of the script used for a spoken text.

<segmentation> describes the principles according to which the text has been segmented, for example into sentences, tone-units, graphemic strata, etc.

<sourceDesc> supplies a bibliographic description of the copy text(s) from which an electronic text was derived or generated.

<stdVals> specifies the format used when standardized date or number values are supplied.

<textClass> groups information which describes the nature or topic of a text in terms of a standard classification scheme, thesaurus, etc.

<textDesc> provides a description of a text in terms of its *situational parameters*.

All of the above elements may be multiply defined within a single header, that is, there may be more than one instance of any declarable element type at a given level. When this occurs, the

following rules apply:

- every declarable element must bear a unique identifier
- for each different type of declarable element which occurs more than once within the same parent element, exactly one element must be specified as the default

In the following example, an editorial declaration contains two possible <correction> policies, one identified as C1 and the other as C2. Since there are two, one of them (in this case C1) must be specified as the default:

```
<editorialDecl>
    <correction id=C1 default=Y> ... </correction>
    <correction id=C2> ... </correction>
    <normalization id=N1>
    <p> ...
    <p> ...
</editorialDecl>
```

For texts associated with the header in which this declaration appears correction method C1 will be assumed, unless they explicitly state otherwise. Here is the structure for a text which does state otherwise:

```
<text>
    ...
    <div1 n=d1> ... </div1>
    <div1 n=d2 decls=C2> ... </div1>
    <div1 n=d3> ... </div1>
    ...
</text>
```

In this case, the contents of the divisions D1 and D3 will both use correction policy C1, and those of division D2 will use correction policy C2.

The decls attribute is defined for any element which is a member of the class *declaring*. This includes the major structural elements <text>, <group>, and , as well as smaller structural units, down to the level of paragraphs in prose, individual utterances in spoken texts, and entries in dictionaries. However, TEI recommended practice is to limit the number of multiple declarable elements used by a document as far as possible, for simplicity and ease of processing.

The identifier or identifiers specified by the decls attribute are subject to two further restrictions:

- An identifier specifying an element which contains multiple instances of one or more other elements should be interpreted as if it explicitly identified the elements identified as the default in each such set of repeated elements
- Each element specified, explicitly or implicitly, by the list of identifiers must be of a different type.

To demonstrate how these rules operate, we now expand our earlier example slightly:

```
<encodingDesc>
       ...
       <editorialDecl  id=ED1 default=Y>
           <correction id=C1a default=Y>
           <correction id=C1b>
           <normalization id=N1>
           <p> ...
           <p> ...
       </editorialDecl>
       <editorialDecl id=ED2>
           <samplingDecl id=SAMP2>
           <correction id=C2a default=Y>
           <correction id=C2b>
           <normalization id=N2a>
           <normalization id=N2b default=Y>
```

```
        <p>  <p>
      </editorialDecl>
      ...
</encodingDesc>
```

This encoding description now has two editorial declarations, identified as ED1 (the default) and ED2. For texts not specifying otherwise, ED1 will apply. If ED1 applies, correction method C1a and normalization method N1 apply, since these are the specified defaults within ED1. In the same way, for a text specifying decls as "ED2", correction C2a, sampling SAMP2 and normalization N2b will apply.

A finer grained approach is also possible. A text might specify <text decls='C2b N2a'>, or even <text decls='C1a N2a SAMP2'>, to "mix and match" declarations as required. A tag such as <text decls='ED1 ED2'> would (obviously) be illegal, since it includes two elements of the same type; a tag such as <text decls='ED2 C1a'> is also illegal, since in this context "ED2" is synonymous with the defaults for that editorial declaration, namely "SAMP2 C2a N2b", resulting in a list that identifies two correction elements (C1a and C2a).

23.3.3 Summary

The rules determing which of the declarable elements are applicable at any point may be summarized as follows:

1. If there is a single occurrence of a given declarable element in a corpus header, then it applies by default to all elements within the corpus.
2. If there is a single occurrence of a given declarable element in the text header, then it applies by default to all elements of that text irrespective of the contents of the corpus header.
3. Where there are multiple occurrences of declarable elements within either corpus or text header,
   - each must have a unique value specified as the value of its id attribute;
   - one only must bear a default attribute with the value YES.
4. It is a semantic error for an element to be associated with more than one occurrence of any declarable element.
5. Selecting an element which contains multiple occurrences of a given declarable element is semantically equivalent to selecting only those contained elements which are specified as defaults.
6. An association made by one element applies by default to all of its descendants.

23.4 Linguistic Annotation of Corpora

Language corpora often include analytic encodings or annotations, designed to support a variety of different views of language. The present Guidelines do not advocate any particular approach to linguistic annotation (or "tagging"); instead a number of general analytic facilities are provided which support the representation of most forms of annotation in a standard and self-documenting manner. Analytic annotation is of importance in many fields, not only in corpus linguistics, and is therefore discussed in general terms elsewhere in the Guidelines.[4] The present section presents informally some particular applications of these general mechanisms to the specific practice of corpus linguistics.

23.4.1 Levels of Analysis

By *linguistic annotation* we mean here any annotation determined by an analysis of linguistic features of the text, excluding as borderline cases both the formal structural properties of the text (e.g. its division into chapters or paragraphs) and descriptive information about its context

[4] See in particular chapters 14 ('Linking, Segmentation, and Alignment') on p. 331, 15 ('Simple Analytic Mechanisms') on p. 381, and 16 ('Feature Structures') on p. 397.

(the circumstances of its production, its genre or medium). The structural properties of any TEI-conformant text should be represented using the structural elements discussed elsewhere in this chapter and in chapters 6 ('Elements Available in All TEI Documents') on p. 119, 7 ('Default Text Structure') on p. 183, and the various chapters of Part III (on base tag sets). The contextual properties of a TEI text are fully documented in the TEI Header, which is discussed in chapter 5 ('The TEI Header') on p. 77, and in section 23.2 ('Contextual Information') on p. 540 of the present chapter.

Other forms of linguistic annotation may be applied at a number of levels in a text. A code (such as a word-class or part-of-speech code) may be associated with each word or token, or with groups of such tokens, which may be continuous, discontinuous or nested. A code may also be associated with relationships (such as cohesion) perceived as existing between distinct parts of a text. The codes themselves may stand for discrete non-decomposable categories, or they may represent highly articulated bundles of textual features. Their function may be to place the annotated part of the text somewhere within a narrowly linguistic or discoursal domain of analysis, or within a more general semantic field, or any combination drawn from these and other domains.

The manner by which such annotations are generated and attached to the text may be entirely automatic, entirely manual or a mixture. The ease and accuracy with which analysis may be automated may vary with the level at which the annotation is attached. The method employed should be documented in the <interpretation> element within the encoding description of the TEI Header, as described in section 5.3.3 ('The Editorial Practices Declaration') on p. 96. Where different parts of a corpus have used different annotation methods, the decls attribute may be used to indicate the fact, as further discussed in section 23.3 ('Associating Contextual Information with a Text') on p. 550.

An extended example of one form of linguistic analysis commonly practised in corpus linguistics is given in section 15.4 ('Linguistic Annotation') on p. 392.

23.5 Recommendations for the Encoding of Large Corpora

These Guidelines include proposals for the identification and encoding of a far greater variety of textual features and characteristics than is likely to be either feasible or desirable in any one language corpus, however large and ambitious. The reasoning behind this catholic approach is further discussed in chapter 1 ('About These Guidelines') on p. 5. For most large scale corpus projects, it will therefore be necessary to determine a subset of TEI recommended elements appropriate to the anticipated needs of the project. Mechanisms for tailoring the TEI dtd to implement such a subset are described in chapter 3 ('Structure of the TEI Document Type Definition') on p. 35 and chapter 29 ('Modifying the TEI DTD') on p. 619; they include the ability to exclude selected element types, add new element types, and change the names of existing elements. A discussion of the implications of such changes for TEI conformance is provided in chapter 28 ('Conformance') on p. 611.

Because of the high cost of identifying and encoding many textual features, and the difficulty in ensuring consistent practice across very large corpora, encoders may find it convenient to divide the set of elements to be encoded into the following three categories:

required texts included within the corpus will always encode textual features in this category, should they exist in the text

recommended textual features in this category will be encoded wherever economically and practically feasible; where present but not encoded, a note in the header should be made.

optional textual features in this category may or may not be encoded; no conclusion about the absence of such features can be inferred from the absence of the corresponding element in a given text.

Part V

Auxiliary Document Types

Chapter 24

The Independent Header

Many libraries, text repositories, research sites and related institutions collect bibliographic and documentary information about machine readable texts without necessarily collecting the texts themselves. Such institutions may thus want access to the header of a TEI document without its attached text in order to build catalogues, indexes and databases that can be used by people to locate relevant texts at remote locations, obtain full documentation about those texts, and learn how to obtain them. This chapter of the Guidelines describes a set of practices by which the headers of TEI documents can be extracted from those documents and exchanged as freestanding independent TEI documents. Headers exchanged independently of the documents they describe are called *independent headers*.

This chapter outlines practices recommended for encoders (especially those responsible for the documentation of text) when creating independent headers to be distributed, and specifies the set of recommended elements that should be included in the independent header. Of interest to librarian cataloguers who may receive independent headers from remote sites, it also discusses the relationship between the elements of TEI headers and MARC tags, in order to facilitate the cataloguing of these headers or the loading of independent headers into local MARC-based bibliographic databases. This chapter does *not* describe how to create a header. Guidance on the creation of headers and descriptions of each element in the header can be found in chapter 5 ('The TEI Header') on p. 77.

24.1 Definition and Principles for Encoders

An *independent header* is a header extracted from a TEI text that can be exchanged as an independent document between libraries, archives, collections, projects, and individuals. The file description of the independent header (enclosed by the <fileDesc> element) can be used to generate bibliographic records. The profile description, encoding description, and revision history (encoded by the <profileDesc>, <encodingDesc>, and <revisionDesc> elements) can form part of a bibliographic description or, more appropriately, be used as an attached "codebook" for full documentation of the analysis of the text and how it was encoded. Thus, the independent header can serve as the primary means by which libraries, archives, related repositories, research projects, and individual researchers can obtain bibliographic, descriptive, and full documentary information on machine-readable texts that reside in remote locations.

The structure of an independent header is exactly the same as that of a <teiHeader> attached to a document, and can therefore be validated using the same document type definition (DTD). In practice, this means that a <teiHeader> and its DTD can be extracted from a TEI document and shipped to a receiving institution with little or no change. However, some fields that are listed as "optional" in the header are listed as "recommended" for the independent header. For this reason, this chapter should be consulted in connection with any plan to send headers as independent documents.

When deciding which information to include in the independent header, and the format or

structure of that information, the following should be kept in mind:

The independent header should provide full bibliographic information on the encoded text, its source, where the text can be located, and any restrictions governing its use.

The independent header should contain useful information about the encoding of the text itself. In this regard, it is highly recommended that the encoding description be as complete as possible. The Guidelines do not require that the encoding description be included in the header (since some simple transcriptions of small items may not require it), but in practice the use of a header without an encoding description would be severely limited.

The independent header should be amenable to automatic processing, particularly for loading into databases and for the creation of publications, indexes, and finding aids, without undue editorial intervention on the part of the receiving institution. For this reason, two recommendations are made regarding the format or structure of the header: first, where there is a choice between a prose content model and one that contains a formal series of specialized elements, *wherever possible and appropriate the specialized elements should be preferred to unstructured prose*. For instance, the source description can contain either a free-prose citation (tagged <bibl> or even <p>) or a <biblStruct> element, which provides a more rigorous structure for the bibliographic information (see examples in section 6.10 ('Bibliographic Citations and References') on p. 162). The more structured <biblStruct> element is more suitable for automatic processing, and is therefore recommended over the less structured alternatives whenever the header is to be exchanged as an independent header. Second, with respect to corpora, information about each of the texts within a corpus should be included in the overall corpus-level <teiHeader>. That is, source information, editorial practices, encoding descriptions, and the like should be included in the relevant sections of the corpus <teiHeader>, with pointers to them from the headers of the individual texts included in the corpus. There are three reasons for this recommendation: first, the corpus-level header will contain the full array of bibliographic and documentary information for each of the texts in a corpus, and thus be of great benefit to remote users, who may have access only to the independent header; second, such a layout is easier for the coder to maintain than searching for information throughout a text; and third, generally speaking, this practice results in greater overall consistency, especially with respect to bibliographic citations.

24.2 Required and Recommended Tags

The richness and size of the header reflect the diversity of uses to which electronic texts conforming to these Guidelines will be put. It is not intended, however, that all of the elements recommended in this chapter be present in every header. As described in section 5.6 ('Minimal and Recommended Headers ') on p. 114, the TEI header allows for the provision of a very large amount of information concerning the text itself, its source, encodings, and revisions as well as detailed descriptive information that can be used by researchers in analysing the text. The amount of encoding will depend on the nature and intended use of the text. At one extreme, an encoder may expect that the header will only provide bibliographic information about the text adequate to local needs. At the other, wishing to ensure that their texts can be used for the widest range of applications, encoders will want to document as explicitly as possible both bibliographic and descriptive information in such a way that no prior or ancillary knowledge about the text is needed in order to process it. The header, in the latter case, will be very full, approximating the kind of documentation often supplied in the form of a manual. Most texts will lie somewhere between these extremes; textual corpora in particular will tend toward the latter extreme.

The following is a list of the components of the header, in the order in which they are presented in chapter 5 ('The TEI Header') on p. 77, together with an indication of their importance in constructing an independent header.

- <fileDesc> required. Some subelements are required, others optional or recommended:
  - <titleStmt> required; subelements are required or optional:
    <title> required
    <author> required, if known

<sponsor> optional
<funder> optional
<principal> required, if known
<resp> required, if known
<role> and <name> required, if known, when the responsibility is not an author, sponsor, funding body, or principal researcher. Details may be found in section 5.2.1 ('The Title Statement') on p. 82.

- <editionStmt> recommended
  <edition> recommended
  <resp> recommended
  <role> and <name> recommended primarily to distinguish editions.
- <extent> optional
- <publicationStmt> required
  <date> recommended
  <publisher>, <distributor>, or <authority> required
  <pubPlace> recommended
  <address> recommended; prose is sufficient
  <idno> recommended
  <availability> recommended
- <seriesStmt> optional
  <title> required
  <idno> recommended
  <resp> and <name> optional
- <notesStmt> recommended

- <sourceDesc> required. As much information as possible should be provided to identify the source. The following tags are either required or recommended, but other tags not listed here should be used wherever applicable in order to provide an accurate identification of the source. In some instances, the <biblFull> tag is preferable to the <biblStruct> tag.
  - <biblStruct> recommended. For a full discussion of <biblStruct>, see section 6.10 ('Bibliographic Citations and References') on p. 162.
  - <analytic> required when the citation describes an item within a larger collection, such as an essay within a collection or an article in a journal, and is not an independent publication. If used, it should contain the following elements in this order:
    <author> required, if known
    <title> required
    <editor> recommended
  - <monogr> mandatory when applicable; this element should contain the following elements in this order:
    <author> required, if known.
    <title> required. The level attribute must be used to indicate whether this is the title of a book, journal, or series. It is highly recommended that the type attribute be used to distinguish the main title from subordinate, parallel, or other titles. All elements that indicate intellectual responsibility for a work, such as <editor>, are required, if known.
    <imprint> required.
    <pubPlace> required, if known.
    <org> recommended.
    <date> required. If the date is unknown, n.d. may be used.
    <idno> recommended.
    <series> required, if the item is part of a series.
    <title> required, but type attribute is optional.
  - <scriptStmt> required for transcribed speech. See section 5.2.9 ('Computer Files Composed of Transcribed Speech ') on p. 91.
  - <recordingStmt> mandatory when applicable:

<resp> and <name> recommended
<recording> recommended
<equipment> recommended
<broadcast> recommended
<comment> optional

- <encodingDesc> very highly recommended, especially for projects, collections, or corpora. If the <encodingDesc> element is used, it is recommended that it contain one or more of the following elements, rather than a prose description. See section 5.3 ('The Encoding Description') on p. 93.
  - <projectDesc> optional
  - <samplingDecl> optional
  - <editorialDecl> recommended; it is also recommended that the editorial declaration make use of the specialized elements defined in section 5.3.3 ('The Editorial Practices Declaration') on p. 96, rather than only consisting of prose paragraphs. Prose may of course be used in addition to these elements for material otherwise not handled.
  - <tagsDecl> recommended
  - <refsDecl> optional in general, but recommended if a standard referencing system is built into the encoded works. Section 5.3.5 ('The Reference System Declaration ') on p. 100 describes three different methods for documenting the referencing system: the prose method, the stepwise method, and the milestone method. No preference is expressed for one type of method over another, since this depends on the convenience of the encoder and the likely efficiency of the particular software applications envisaged for the text. Only one method can be used within a single <refsDecl> element. If a text uses both hierarchical and milestone tagging, this can only be described in prose.
  - <classDecl> required where the scheme attribute has been used to identify the classification scheme or taxonomy used by any of the elements <keywords>, <classcode>, <occupation> or <socecstatus>. Even where this is not done, this element may usefully document the classification employed, either explicitly as a series of <taxonomy> elements, or implicitly by means of bibliographic citation.
- <profileDesc> recommended
  - <langUsage> recommended
  - <language> recommended
- <textDesc> optional in most instances, but recommended when the encoder wants to provide a full description of the situation within which a text was produced or experienced, characterize it in a relatively continuous manner (in contrast to discrete categories based on type or topic), and believes that this characterization of the text will be helpful to the understanding, analysis, or retrieval of this text by remote users. If a collection or corpus uses a pre-existing descriptive typology as its organizing principle, it is recommended that its components be re-expressed in terms of the parameters listed here. If the encoder believes that pre-existing text categories (such as a standard classification scheme) are sufficient, then it is recommended that the <textClass> element be used instead. See section 23.2.1 ('The Text Description') on p. 541 for details and guidance.
  - <channel> required
  - <constitution> required
  - <derivation> required
  - <domain> required
  - <factuality> required
  - <interaction> required
  - <preparedness> required
  - <purposes> required
  - <purpose> required
- <textClass> optional in most instances; this element may may be used as an alternative or in addition to the <textDesc> element. <textClass> is recommended in the following situations:

  - a standard text category, such as the Library of Congress List of Subject Headings or a Dewey Decimal Classification category, clearly describes the text
  - situational parameters (or the demographic elements of the <particDesc> element) are used and a text category can be constructed by the encoder based on a recurring set of values for those parameters.

  See section 5.4.3 ('The Text Classification') on p. 111 for details and guidance. One or more of the following sub-elements can be used.

  - <keywords> recommended only if using a standard thesaurus such as the Library of Congress List of Subject Headings, a discipline-specific thesaurus, or a thesaurus defined explicitly in the header. In each case, the source should be indicated by the scheme attribute and defined in the <classDecl> element.
  - <classCode> recommended only if the text is categorized by an internationally accepted classification scheme, such as the Dewey Decimal or Universal Decimal classification schemes. The scheme should be indicated by the scheme attribute and defined in the <classDecl> element.
  - <catRef> optional in most instances, but recommended when a user-defined classification is in use. The scheme should be indicated by the scheme attribute and defined in the <classDecl> element.

- <particDesc> optional, but recommended for spoken text when the encoder judges that such information is useful to remote users in the analysis of that text, and for both written and spoken text if such information is useful in the analysis of language usage. For details and guidance, see section 23.2.2 ('The Participants Description') on p. 545.
- <participant> or <particGroup> recommended. Though the substructure of both the <participant> and <particGroup> elements can be prose, in independent headers one or more of the following sub-elements providing more specific details should be used in preference to prose. Users of these Guidelines are free to extend the set of headings listed below.

  - <name> recommended when the information is available
  - <birthDate> recommended when the information is available
  - <birthPlace> recommended when the information is available
  - <firstLang> recommended when the information is available
  - <langKnown> recommended when the information is available
  - <residence> recommended when the information is available
  - <education> recommended when the information is available
  - <affiliation> recommended when the information is available
  - <occupation> it is recommended that, where possible, the classification of the trade, occupation, or profession be derived from a standard classification or taxonomy, and that the source taxonomy be identified in the scheme attribute.
  - <socecstatus> it is recommended that, where possible, the encoding of social and economic status be derived from a standard classification or taxonomy, and that the source taxonomy be identified in the scheme attribute.

- <particRelations> optional, but recommended where it is judged by the encoder that such information is important to the analysis of the text. If the <particRelations> tag is used, it is recommended that the special purpose <relation> element be used. See section 23.2.2 ('The Participants Description') on p. 545.
- <settingDesc> optional, but recommended when the encoder judges that this information is useful in the analysis of the text, particular in the analysis of language usage.
- <revisionDesc> required in the independent header when available. It is recommended that the <revisionDesc> be encoded with a series of special purpose <change> elements with grouped <name>, <what> and <date> tags.

24.3 Header Elements and their Relationship to the MARC Record

This section offers some guidance to both cataloguers and bibliographic analysts who want to load TEI independent headers into a MARC-based retrieval system. Because there are variations in cataloguing practice across local sites, among bibliographic utilities (such as OCLC and RLIN), and differences in MARC usage in different countries, only tentative advice is possible. Note that the following examples are based on USMARC, *not* UNIMARC. [1] UNIMARC offers cataloguers in different countries the opportunity to combine different national practices in a single MARC format, and is the preferred variety of MARC records for distribution across national boundaries. The implementation of UNIMARC, however, will be affected by local practice and by guidelines offered by the bibliographic utilities. Though UNIMARC is a stable format, the guidelines for its implementation are not sufficiently known or stabilized to be included in this chapter.

There are some major differences between the MARC record and the TEI header that will cause problems for librarians trying to map from the TEI independent header to the MARC record. The most important difference between the MARC record and the TEI header is the function of each. Despite the efforts and claims of some members of the library community, the MARC record remains fundamentally an electronic version of the catalogue card, with the limitations of its model. [2] The catalogue card is a unitary record for a physical object containing complex *bibliographic data* of varying sorts. The catalogue card points to the physical object. The TEI header provides full bibliographic information (as would a card), as well as documentary non-bibliographic information that supports the analysis, either by humans or machines, of the electronic text documented by header. Most of this analytical information, which is found in profile description, encoding description, and revision history, has little direct provision for it in the MARC record, and if retained must be recorded as unstructured notes (55XX) fields. Notes fields usually do not have the structure to support machine retrieval and analysis, while properly formatted profile, encoding, and revision descriptions lend themselves to retrieval, can support machine processing (including analysis), and point directly to the electronic text attached to the header. Moreover, the electronic text points back to the relevant elements in the header.

Though this chapter offers some advice on where the profile, encoding, and revision descriptions might go in a MARC record, for practical reasons a repository might want create a codebook from these divisions of the header, and create a MARC record from the file description only. The MARC record should contain a reference to the codebook.

Subfields (or delimiters) are conventionally indicated by the dollar sign.

24.4 MARC Fields for the File Description

Note that there is no provision for the "Main Entry" (or USMARC 1XX fields) in the TEI header. The main entry should be constructed, using appropriate name authority control, by the cataloguer from information derived from the header that indicates who is primarily responsible for the intellectual content of the work. There is an <author> tag, but the form of the name will have to be checked by a cataloguer before the main entry is constructed.

- <titleStmt> corresponds to title and statement of responsibility fields in MARC, typically 240 (for uniform title) and 245 (for title proper).
- <title> 240 $a (for uniform titles) or 245 $a fields. Put any subtitles in 24X $b. Insert the constant, "[computer file]" in the 24X $h gmd subfield.

[1] For more information on UNIMARC, see Brian P. Holt, *UNIMARC Manual* (London, U.K.: IFLA Universal Bibliographic Control and International MARC Programme, British Library, 1987). For USMARC, see Walt Crawford, *MARC for library use: understanding USMARC* (Boston: G.K. Hall, 1989), *USMARC format for bibliographic data, including content designation* (Washington, D.C.: Library of Congress, 1987), and Deborah J. Byrne, *MARC manual : understanding and using MARC records* (Englewood, Colo.: Libraries Unlimited, Inc., 1991).

[2] The primary function of the MARC record when it was first designed in the mid-1960s was to allow for the electronic distribution of cataloguing records in support of card production. See Henriette Avram, *The MARC Pilot Project* (Washington D.C.: Library of Congress, 1968), p. 3. For discussion of the relationship between the MARC record and the catalogue card, see Michael Gorman, "After AACR2R: The Future of the Anglo-American Cataloging Rules," in Richard Smiraglia, ed., *Origins, Content and Future of AACR2 Revised* (Chicago: American Library Association, 1992).

The following elements belong in the 245 $c subfield: statement of responsibility.

- <sponsor>
- <funder>
- <principal>

Example:

```
<titleStmt>
  <title>Two stories by Edgar Allen Poe: electronic
         version</title>
  <author>Poe, Edgar Allen (1809-1849)</author>
  <respStmt><resp>compiled by</resp>
  <name>James D. Benson</name></respStmt>
```

This might be tagged in MARC as:

```
245 Two stories by Edgar Allen Poe :$belectronic version ;
    compiled by $cJames D. Benson.
```

- <edition> 250 $a
- <name> 250 $b

Example:

```
<editionStmt><edition>
    Student's edition,
    <date>June 1987</date>
</edition><respStmt>
    <resp>New annotation by</resp>
    <name>George Brown</name>
</respStmt></editionStmt>
```

This might be tagged in MARC as:

```
250  $aStudent's edition, June, 1987, new annotation by
     $bGeorge Brown.
```

- <extent>. The extent is analogous to the "Physical Description" MARC field. Fields 256 or 3XX, depending on local practice are appropriate.
- <date> 260 $c, and appropriate fixed fields.
- <publisher>, <distributor>, or <authority> 260 $b
- <pubPlace> 260 $a

Example:

```
<publicationStmt>
  <publisher>Columbia University Press</publisher>
  <pubPlace>New York</pubPlace>
  <date>1993</date>
</publicationStmt>
```

This may be tagged in MARC as:

```
260 $aNew York :$bColumbia University Press, $c1993.
```

Local practice will determine appropriate MARC fields for <address>, <idno>, and <availability>. Restrictions on access should normally be placed in the 506 field, while the place where an item may be ordered will be located in a local notes (590) field. If local practice warrants it, the address of the publisher should be indicated in the 260 field.

The series <title> and the <idno> should be placed in the appropriate 490 fields (series untraced), if series authority checking needs to be done. Further, because the TEI tags do not differentiate between name, conference, or title series, there is no simple mechanical method for determining which MARC tag (410, 411, etc.) should be used. Safe practice would be to load any series statements into 490 fields, and then to conduct authority work on those fields.

- <notesStmt> These are usually reserved for general notes (500) fields.

The <sourceDesc> can be mapped to be a "source of data" note (537 in RLIN MDF format) with the print constant "Transcribed from:" at the beginning of the note. The <biblStruct> itself can be mapped onto a 581 field (note on primary publication) using the ISBD format to separate each data element.

The <scriptStmt>, <recordingStmt>, <recording>, <equipment>, and <broadcast> elements do not easily map to existing MARC fields, and should be put into a local notes field (590) treating the TEI tag introducing each component as a print constant at the head of the field in order to facilitate future local processing and retrieval.

Example:

```
<scriptStmt id=CNN12>
  <bibl><author>CNN Network News</author>
    <title>News Headlines</title>
    <date>12 Jun 1991</date>
  </bibl>
</scriptStmt>
```

This may be tagged in MARC thus:

```
590  <scriptStmt id=CNN12>
     <bibl><author>CNN Network News</author>
     <title>News Headlines</title>
     <date> 12 Jun 1991></date></bibl>
     </scriptStmt>
```

Example:

```
<recordingStmt>
   <recording type=video dur="10 mins">
   <equipment><p>Recorded from FM radio to chrome
       tape</p></equipment>
   <broadcast>
      <bibl>
         <title>Britain's pleasure parade</title>
         <author>BBC Radio 4 FM</author>
         <editor role=interviewer>Robin Day</editor>
         <editor role=interviewee>Margaret Thatcher</editor>
         <series><title>The World Tonight</></series>
         <date>27 Nov 89</date>
       </bibl>
    </broadcast>
    </recording>
</recordingStmt>
```

This can be tagged in MARC as:

```
590 <recordingStmt>
    <recording type=video dur="10 mins">
    <equipment><p>Recorded from FM radio to chrome
         tape</p></equipment><broadcast>
    <bibl><title>Britain's pleasure parade</title>
    <author>BBC Radio 4 FM</author>
    <editor role=interviewer>Robin Day</editor>
    <editor role=interviewee>Margaret Thatcher</editor>
    <series><title>The World Tonight</></series>
    <date>27 Nov 89</date>
    </bibl>
    </broadcast>
    </recording>
    </recordingStmt>
```

24.5 MARC Fields for the Encoding Description

The <encodingDesc> element provides useful information documenting the relationship between an electronic text and the source or sources from which it was derived. The <projectDesc>, <samplingDecl>, <editorialDecl>, and <refsDecl> elements provide details of decisions and rationales used about the process and purposes of the project, how text was sampled, principles of editorial practice, and how canonical references are constructed. The 567 field (notes on methodology) appears to be the most appropriate for this sort of information, though this field is normally intended for methodologies characterizing the social sciences. Practically, it would be wise to transcribe the <projectDesc>, <editorialDecl>, <refsDecl>, and <classDecl> elements directly as one or more 567 fields without intervention, with the element name at the beginning of each field, and any TEI tags left intact. This may facilitate any locally-developed retrieval software.

Example:

```
<encodingDesc>
  <projectDesc><p>Texts were collected to illustrate the
    full range of twentieth-century spoken and written Swedish,
    written by native Swedish authors.</projectDesc>
  <samplingDecl><p>Sample of 2000 words taken from the
    beginning of the text.</p></samplingDecl>
  <editorialDecl>
       <interpretation><p>Errors in transcription controlled
         by using the SUC spell checker, v.2.4</p></interpretation>
  </editorialDecl>
</encodingDesc>
```

This may be tagged in MARC as:

```
567  <projectDesc><p>Texts were collected to illustrate the
      full range of twentieth-century spoken and written
      Swedish, written by native Swedish authors.</p>
567  <samplingDecl><p>Sample of 2000 words taken from the
     beginning of the text.</p>
567  <editorialDecl>
     <interpretation><p>Errors in transcription controlled
      by using the SUC spell checker, v. 2.4</p>
     </interpretation>
     </editorialDecl>
```

24.6 MARC Fields for the Profile Description

The profile description is the most problematic element in the TEI header for librarian cataloguers, because it provides a detailed description of the *non-bibliographic* aspects of the text, specifically the languages and sublanguages used, the situation in which it was produced, and the participants and their setting. This information can be used for retrieval purposes or in machine-supported analysis of the text. The information can be loaded into a separate "codebook" and referenced by the MARC record. Little guidance can be offered on the appropriate MARC location for the elements that make up the profile description, except to suggest that if a site wants to load the profile description into a MARC record for archival and possibly retrieval purposes, then the contents of the profile description may be mapped into a locally-defined notes field (59X) with its TEI tags intact, as in the examples above.

24.7 MARC fields for the Revision Description

The revision history (<revisionDesc>) logs all changes to a machine readable file whether or not these constitute a new edition of the file. Aside from the edition area of the MARC record,

there are no MARC fields that deal specifically with changes of this sort. This information might be best included in a "codebook", rather than a MARC record. As before, the simplest way of approaching this problem is to include the material with its TEI tags intact as a locally-defined note (59X) in order to support future local processing.

24.8 Structure of the DTD for Independent Headers

The following document type definition is provided in file *teishd2.dtd* and constitutes the auxiliary DTD for independent headers as described in this chapter.

```
<!-- 24.8: File teishd2.dtd: Auxiliary DTD for Independent    -->
<!-- Header                                                   -->
<!-- Text Encoding Initiative: Guidelines for Electronic      -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.    -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy    -->
<!-- in any form is granted, provided this notice is          -->
<!-- included in all copies.                                  -->

<!-- These materials may not be altered; modifications to     -->
<!-- these DTDs should be performed as specified in the       -->
<!-- Guidelines in chapter "Modifying the TEI DTD."           -->

<!-- These materials subject to revision. Current versions    -->
<!-- are available from the Text Encoding Initiative.         -->

<!-- Embed entities for TEI generic identifiers.              -->

<!ENTITY % TEI.elementNames system 'teigis2.ent'               >
%TEI.elementNames;

<!-- Embed entities for TEI keywords.                         -->

<!ENTITY % TEI.keywords.ent system 'teikey2.ent'               >
%TEI.keywords.ent;

<!-- Define element classes for content models, shared        -->
<!-- attributes for element classes, and global attributes.   -->
<!-- (This all happens within the file teiclas2.ent.)         -->

<!ENTITY % TEI.elementClasses system 'teiclas2.ent'            >
%TEI.elementClasses;

<!-- Now declare the IHS element.                             -->

<!ELEMENT ihs           - O  (teiHeader+)                      >
<!ATTLIST ihs                %a.global;                        >

<!-- Finally, embed the TEI header and core tag sets.         -->

<!ENTITY % TEI.header.dtd system 'teihdr2.dtd'                 >
%TEI.header.dtd;
<!ENTITY % TEI.core.dtd system 'teicore2.dtd'                  >
%TEI.core.dtd;
```

The overall structure of a set of independent headers, encoded for interchange as a group, is thus:

```
<!DOCTYPE ihs system 'teishd2.dtd'>
<ihs>
```

```
<teiHeader>
  <fileDesc> ...     </fileDesc>
  <encodingDesc> ... </encodingDesc>
  <profileDesc> ...  </profileDesc>
  <revisionDesc> ... </revisionDesc>
</teiHeader>
<teiHeader>
  <fileDesc> ...     </fileDesc>
  <encodingDesc> ... </encodingDesc>
  <profileDesc> ...  </profileDesc>
  <revisionDesc> ... </revisionDesc>
</teiHeader>
<teiHeader> ... </teiHeader>
<!-- ... etc. -->
</ihs>
```

In practice, headers might be stored in separate operating system files, to reduce redundant storage requirements; in this case, the top-level file for a typical document might have the following structure:

```
<!DOCTYPE tei system 'tei2.dtd' [
  <!ENTITY txt01 system 'text01.tei' >
  <!ENTITY hdr01 system 'text01.hdr' >
]>
<tei.2>
&hdr01
&txt01
</tei.2>
```

while that for a set of independent headers might have this structure:

```
<!DOCTYPE ihs system 'teishd2.dtd' [
  <!ENTITY hdr01 system 'text01.hdr' >
  <!ENTITY hdr02 system 'text02.hdr' >
  <!ENTITY hdr03 system 'text03.hdr' >
  <!-- ... etc. -->
]>
<ihs>
&hdr01
&hdr02
&hdr03
<!-- etc. -->
</ihs>
```

Chapter 25

Writing System Declaration

The *writing system declaration* or WSD is an auxiliary document which provides information on the methods used to transcribe portions of text in a particular language and script. We use the term *writing system* to mean a given method of representing a particular language, in a particular script or alphabet; the WSD specifies one method of representing a given writing system in electronic form. A single WSD thus links three distinct objects:

- the language in question
- the writing system (script, alphabet, syllabary) used to write the language
- the coded character set, entity names, or transliteration scheme used to represent the graphic characters of the writing system

Different natural languages thus have different writing system declarations, even if they use the same script. Different methods used to write the same language (e.g. Cyrillic or Latin encoding of Serbo-Croatian), and different methods of representing the same script in electronic form (e.g. different coded character sets such as ASCII or EBCDIC, or different transliteration schemes) similarly must use different writing system declarations.

This chapter describes first the overall structure of the WSD (section 25.1 ('Overall Structure of Writing System Declaration') on p. 571), and then the specific elements used to document the natural language, writing system, coded character sets, SGML entity names, and transliteration schemes united by the WSD. Section 25.6 ('Linkage between WSD and Main Document') on p. 582 describes how the WSD is associated with different portions of a document. Predefined TEI writing system declarations, which should suffice for many uses, are described in section 25.7 ('Predefined TEI WSDs') on p. 582. There follows a brief description of how to create a new WSD on the basis of an existing WSD. Finally, in section 25.8 ('Details of WSD Semantics') on p. 583 we provide a formal discussion of the semantics of the writing system declaration.

25.1 Overall Structure of Writing System Declaration

A writing system declaration is a distinct auxiliary document, separate from any transcription for which it is used. A TEI document specifies the writing system declarations applicable to it by means of declarations in its header, and by means of its use of the global **lang** attribute, as further specified in section 25.6 ('Linkage between WSD and Main Document') on p. 582. Each writing system declaration is itself a free-standing document, encoded as a single **<writingSystemDeclaration>** element, and containing the following elements:

<writingSystemDeclaration> declares the coded character set, transliteration scheme, or entity set used to transcribe a given writing system of a given language. Attributes include:

name gives a formal name for the writing system declaration

date gives the date on which the writing system declaration was last revised.

<language> identifies the language being described in the writing system declaration.

<script> contains a prose description of the script declared by a writing system declaration.

<direction> specifies one or more conventional directions in which a language is written using a given script.

<characters> contains a specification of the characters used in a particular writing system to write a particular language, and of how those characters are represented in electronic form.

<note> (in a writing system) contains a note of any type.

All elements in the writing system declaration may bear either of the following two attributes:

id gives a unique identifier for the element.

lang gives the language in which the content of the element is written.

These attributes function in the same way as the global id and lang attributes of the main TEI DTD (although for technical reasons the latter is declared differently): the former provides a unique SGML identifier for the element, and the latter identifies the language in which the contents of the element are expressed, using a code from ISO 639.

The overall structure of a writing system declaration is thus as follows:

```
<writingSystemDeclaration
          lang='eng'
          name=' ... '
          date='1993-05-29'>

<language iso639='...'>
    <!-- name of language here -->
    </language>
<script>
    <!-- description of script here ... -->
    </script>
<direction chars=LR lines=TB>
<characters>
    <!-- description of character inventory here ... -->
    </characters>
</writingSystemDeclaration>
```

The attributes date and name are required on the <writingSystemDeclaration> element. The date attribute is used to specify the date on which the WSD was written or last changed; this must be given in the format "yyyy-mm-dd", as defined by ISO 8601: 1988, *Data elements and interchange formats — Information interchange — Representation of dates and times*. The name attribute is used to assign a formal name to the writing system declaration, for references to it from elsewhere. It is recommended, though not required, that the name be constructed as an SGML formal public identifier; for purposes of writing system declarations, this means it should follow the pattern of the following examples:

```
-//TEI P2: 1993//NOTATION WSD for Modern English//EN
-//WWP 1993//NOTATION WSD for 17th-century English//EN
-//OTA 1991//NOTATION WSD for Old English//EN
-//GLDV 1997//NOTATION WSD Mittelhochdeutsch//DE
```

That is, the identifier should consist of:

1. either the string -// (indicating that the organization issuing the WSD is not registered with ISO) or the string +// (indicating that the organization is so registered);
2. a short string identifying the issuing organization and optionally the document which defines the WSD (or at least the issuing organization and the date);
3. the separator string //;
4. the keyword NOTATION, indicating that in SGML terms, the writing system declaration functions as a non-SGML notation;
5. a descriptive phrase indicating the contents of the WSD (which may, as here, begin "WSD for ...");
6. the separator string //;
7. the ISO 639 code for the language in which the WSD is written, for example EN.

The other elements of the WSD are described in the following sections.

The DTD for writing system declarations is included in file *teiwsd2.dtd*; a writing system declaration will thus begin with a document type declaration invoking that file:

```
<!DOCTYPE writingSystemDeclaration system 'teiwsd2.dtd' [
  <!-- entity sets are declared and invoked here -->
 ]>
```

The formal declaration of the writing system declaration is as follows:

```
<!-- 25.1:  Writing System Declaration                          -->
<!-- Text Encoding Initiative: Guidelines for Electronic        -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.      -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy      -->
<!-- in any form is granted, provided this notice is            -->
<!-- included in all copies.                                    -->

<!-- These materials may not be altered; modifications to       -->
<!-- these DTDs should be performed as specified in the         -->
<!-- Guidelines in chapter "Modifying the TEI DTD."             -->

<!-- These materials subject to revision. Current versions      -->
<!-- are available from the Text Encoding Initiative.           -->
<!ENTITY % INHERITED '#IMPLIED'                                  >
<!ENTITY % ISO-date 'CDATA'                                      >
<!-- Embed entities for TEI generic identifiers.                -->

<!ENTITY % TEI.wsdNames system 'wdgis2.ent'                      >
%TEI.wsdNames;
<!ENTITY % a.global '
          id                  ID                  #IMPLIED
          lang                CDATA               %INHERITED'    >
<!ELEMENT writingSystemDeclaration
                         - -  (language, script, direction*,
                              characters, note*)                 >
<!ATTLIST writingSystemDeclaration
                              %a.global;
          name                CDATA               #REQUIRED
          date                %ISO-date           #REQUIRED      >
<!-- ... declarations from section 25.2                          -->
<!--     (Language identification)                               -->
<!--     go here ...                                             -->
<!-- ... declarations from section 25.3                          -->
<!--     (Script and writing direction)                          -->
<!--     go here ...                                             -->
<!-- ... declarations from section 25.4.1                        -->
<!--     (Base components)                                       -->
<!--     go here ...                                             -->
<!-- ... declarations from section 25.4.2                        -->
<!--     (Exceptions to the base components)                     -->
<!--     go here ...                                             -->
<!-- ... declarations from section 25.5                          -->
<!--     (Notes)                                                 -->
<!--     go here ...                                             -->
```

25.2 Identifying the Language

The <language> element is used to name the language associated with the WSD. Its iso639 attribute gives the ISO standard code for the language as defined by *ISO 639: 1988. Code for the representation of names of languages*.

<language> identifies the language being described in the writing system declaration. Attributes include:

iso639 gives the standard language code from ISO 639.

If the language in question is not included in the list in ISO 639, the value of the attribute iso639 should be the empty string, as in the following example:

```
<language iso639=''>Various</language>
```

The <language> element should not be confused with the global lang attribute; the element identifies the language whose writing system is being documented, while the attribute identifies the language in which the description is being written. A writing system declaration for classical Greek, for example, which itself is written in English, would have the value eng for the lang attribute on the top-level element, and the value grc for the iso639 attribute on the <language> element:

```
<writingSystemDeclaration
          id='grc.beta'
          lang=eng
          name='-//TEI P2: 1993//NOTATION WSD for TLG Beta Code
               transliteration of ancient greek//EN'
          date='1993-05-29'>

<language iso639='grc'>Classical Greek.  This WSD documents the Beta
transcription code for classical Greek developed by the Thesaurus
Linguae Graecae of the University of California, Irvine.</language>
<!-- ... -->
</writingSystemDeclaration>
```

Normally, the language described is a natural language; in some cases, however, artificial languages, dialects, or other sublanguages may be usefully regarded as a language and documented in a writing system declaration. When a sublanguage is documented, a description of the sublanguage should be included in the <language> element:

```
 <language iso639='jpn'>
    Japanese (specialized writing system for waka)
 </language>
```

When a writing system declaration is prepared solely in order to document a coded character set or entity set suitable for use with many natural languages, the content of the <language> element should be "Various" (or the equivalent in the language of the WSD):

```
<writingSystemDeclaration
          lang=eng
          name='-//TEI P2: 1993//NOTATION WSD for ISO 646 IRV//FR'
          date='1993-05-29'>
  <language iso639=''>Plusieurs</language>
  <!-- ... -->
</writingSystemDeclaration>
```

The <language> element is formally defined thus:

```
<!-- 25.2:  Language identification                          -->
<!ELEMENT language      - O  (#PCDATA)                        >
<!ATTLIST language           %a.global;
          iso639             CDATA              #REQUIRED     >
<!-- This fragment is used in sec. 25.1                      -->
```

25.3 Describing the Writing System

The writing system itself is described in general terms using the following elements:

<script> contains a prose description of the script declared by a writing system declaration.

<direction> specifies one or more conventional directions in which a language is written using a given script. Attributes include:

chars indicates the order in which characters within a line are conventionally presented in this writing system Suggested values include:

LR left to right
RL right to left
TB top to bottom
BT bottom to top

lines indicates the order in which lines conventionally follow each other in this writing system. Suggested values include:

TB top to bottom
BT bottom to top
LR left to right
RL right to left

The <script> element contains a prose description of the script, alphabet, syllabary, or other system of writing used to write the language in question. The <direction> element indicates the direction(s) in which the script is conventionally written. Both these elements are provided for the sake of human readers; neither is likely to be suited to machine processing without human intervention.

The Latin alphabet conventionally used to write English, for example, might be described thus:

```
<script>Latin alphabet (with diacritics for loan words)</script>
<direction chars=LR lines=TB>
```

The chars and lines attributes are used to indicate the direction in which characters within a line, and lines on the page, may legitimately be written using the script in question. If more than one direction is possible, the <direction> element may repeat or its attributes may be given more complex values. A script written vertically top to bottom, with lines arranged either left to right or right to left, for example, might be declared either thus:

```
<direction chars=TB lines=LR>
<direction chars=TB lines=RL>
```

or thus:

```
<direction chars=TB lines='LR RL'>
```

In very complex cases, the attributes may be given prose values:

```
 <direction chars='boustrophedon:  LR, then RL, then LR, etc.'
            lines='TB'>
```

or the element may be omitted entirely (in which case experts on the script should be consulted for advice on proper processing):

```
 <script>Japanese (mixture of kanji, hiragana, katakana)</script>
 <!-- If you don't already know how Japanese is written,
      no DIRECTION element in the world will help you. -->
```

It should be noted that the <direction> element describes conventional display only: all scripts are subject to unusual treatment for aesthetic or other reasons, and such unusual treatment need not be foreseen here. (The Latin alphabet, for example, although conventionally written left-to-right, top-to-bottom, can be set vertically in signs or in other special cases.) Unusual methods of arranging the text on a page are best documented within the document instance by means of the global rend attribute.

The <script> and <direction> elements are declared thus:

```
<!-- 25.3:  Script and writing direction                      -->
<!ELEMENT script        - O  (#PCDATA)                          >
<!ATTLIST script             %a.global;                         >
<!ELEMENT direction     - O  EMPTY                              >
<!ATTLIST direction          %a.global;
          chars              CDATA              #REQUIRED
          lines              CDATA              #REQUIRED       >
<!-- This fragment is used in sec. 25.1                       -->
```

25.4 Documenting the Character Set and Its Encoding

25.4.1 Base Components of the WSD

The characters or graphic symbols of the writing system are documented in the <characters> element of the WSD. This documentation can take any of the following forms:

- reference to an international standard, national standard, or private coded character set
- reference to a public set of SGML entities
- reference to another WSD which documents the same script and the same methods of representing it electronically
- formal declaration of each graphic unit in the writing system
- a combination of the above: reference to one or more standard coded character sets, entity sets, or writing system declarations, followed by individual declaration of all exceptions

The coded character sets, entity sets, and external WSDs referred to are called the *base components* of the writing system declaration. The base components of a WSD are declared within the <characters> element using the following elements:

<characters> contains a specification of the characters used in a particular writing system to write a particular language, and of how those characters are represented in electronic form.

<codedCharSet> identifies a public or private coded character set which is used as a basic component of a writing system declaration.

<baseWsd> identifies a writing system declaration whose mappings among characters, forms, entity names, and bit patterns are to be incorporated (possibly with modifications) in this writing system declaration.

<entitySet> identifies a public or private entity set whose mappings between entity names and characters are to be incorporated (perhaps with modifications) into this writing system declaration.

The elements <codedCharSet>, <baseWsd>, and <entitySet> are all members of the class *baseStandard* and inherit from it the following attributes:

name gives the normal citation form for the standard being referred to.

authority indicates the authority responsible for issuing the standard being referred to: the TEI, the International Organization for Standardization (ISO), a national body, or a private body. Legal values are:

tei the base writing system declaration is a standard WSD issued by the Text Encoding Initiative

iso the character set or entity set was issued by ISO

national the character set or entity set was issued by a national standards body

private the writing system declaration, character set, or entity set was issued publicly by a private organization or project

none the writing system declaration, character set, or entity set has not been publicly issued by any organization; it is specific to an individual text or project

Some simple examples of the use of these elements follow:

```
<codedCharSet name='ANSI X3.4' authority='national'>
<codedCharSet name='ISO 646: 1991' authority='ISO'>
<baseWsd      name='-//TEI P3: 1994//WSD ISO 8859-1//EN'
              authority='TEI'>
<entitySet    name='ISO 8879:1986//ENTITIES Added Latin 1//EN'
              authority='ISO'>
```

The base components identify the set of characters used in the writing system, and further specify, for each character, the string(s) of bytes and entity names used to encode it in the text. This information may be modified by further information given within the <exceptions> element, as described below in section 25.4.2 ('Exceptions in the WSD') on p. 577.

The elements for identifying the base components of the writing system declaration are declared thus:

```
<!-- 25.4.1:  Base components                                    -->
<!ELEMENT characters    - O  (codedCharSet*, baseWsd*,
                              entitySet*, exceptions?)           >
<!ATTLIST characters         %a.global;                          >
<!ENTITY % a.baseStandard '
          name               CDATA               #REQUIRED
          authority          (tei | iso | national | private |
                              none)              #REQUIRED'     >
<!ELEMENT codedCharSet  - O  EMPTY                               >
<!ATTLIST codedCharSet       %a.global;
                             %a.baseStandard;                    >
<!ELEMENT baseWsd       - O  EMPTY                               >
<!ATTLIST baseWsd            %a.global;
                             %a.baseStandard;                    >
<!ELEMENT entitySet     - O  EMPTY                               >
<!ATTLIST entitySet          %a.global;
                             %a.baseStandard;                    >
<!-- This fragment is used in sec. 25.1                          -->
```

25.4.2 Exceptions in the WSD

The <exceptions> element contains definitions for any character which differs in any respect from the specifications contained in the base components of the WSD. If no base components are named, then every character in the writing system must be defined explicitly.

The documentation for each character in the writing system indicates at least the following:

- the string of bytes used to represent the character
- whether the character is a letter, a punctuation mark, a diacritical mark, or falls into some other class
- a brief conventional name or description of the character
- any standard or local entity names used for the character
- the position of the character in the Universal Character Set (UCS) defined by ISO 10646, if known
- the position of the character's form in the tables prepared by the Association for Font Information Interchange

In addition, images of the character encoded in a graphics format or other notation may be associated with the character as internal or external figures.

This information is encoded using the following elements:

<exceptions> documents ways in which a writing system declaration differs from the coded character sets, base writing system declarations, and entity sets which form its bases.

<character> defines one unit in a writing system, supplementing or overriding information provided in the base coded character sets, writing system declarations, and entity sets. Attributes include:

class describes the function of the character using a prescribed classification. Legal values are:

lexical character is used in writing words (lexical items) of the language (includes members of syllabaries and ideographic systems, as well as composite letter-plus-diacritic combinations)

punc character is a punctuation mark which does not appear within lexical items

lexpunc character can appear as a normal punctuation mark, but can also appear within a lexical item (and should usually, when occurring between two lexical characters, be treated as lexical—in English, hyphen and apostrophe are typically treated as members of this class)

digit character is an Arabic decimal numeral (0, 1, ... 9) (does not include superscript numbers, circled numbers, numeric dingbats, etc.)

space character represents some form of white space (space character, horizontal or vertical tab, newline, etc.)

dl character is a diacritic applying to the following lexical character

ld character is a diacritic applying to the preceding lexical character

dia character is a diacritic which is explicitly joined to a lexical character by a joiner character

joiner character is used to join a diacritic to the lexical character to which it applies (in some encoding schemes, the backspace control character may be used as a joiner; in others, a graphic character is used for the same function)

other character does not fall into any of the other classes (dingbats and other unusual characters fall here)

<desc> (in a writing system declaration) contains a description of a character or character form.

<form> identifies one letter form taken by a particular character in a writing system declaration. Attributes include:

string gives the byte string used to encode the letter form in the text.

codedCharSet specifies which base coded character set the string value occurs in.

entityStd gives the name of one or more entities defined for this character form in some standard entity set(s).

entityLoc gives one or more entity names used locally for this character form.

ucs-4 gives the position of the character form in the thirty-two bit "universal character set" defined by ISO 10646.

afiicode gives one or more codes associated with this letter form by the Association for Font Information Interchange.

<figure> (in a writing system declaration) contains an image of a character form, stored in-line in some declared notation. Attributes include:

notation identifies the notation in which the figure is encoded.

<extFigure> (in a writing system declaration) refers to a figure or illustration depicting the character form, which is stored in some declared notation external to the text. Attributes include:

notation identifies the notation in which the figure is stored.

entity gives the SGML name of the external entity which contains the figure.

The <exceptions> element contains a series of <character> elements only, each of which may contain descriptions of the character (including its name), notes, and a series of <form> elements documenting the different forms the character can take. Attributes on the <character> and <form> elements are used to convey the information mentioned above: byte string, entity names, UCS-4 code, etc.

A simple example:

```
<character class=lexical>
  <form string='A' ucs-4='0041' afiicode='0041'>
    <desc>Latin capital letter A</desc>
  </form>
</character>
```

When transliteration schemes are used, the string used to encode the character will typically be in a different alphabet:

```
<character class=lexical>
  <form string='*G' entityStd="Ggr" ucs-4='0393' afiicode='260044'>
    <desc>Greek capital letter Gamma</desc>
  </form>
</character>
```

The UCS-4 code is given as eight hexadecimal digits, one for each four bits of the thirty-two-bit value. For legibility a hyphen may be inserted as a separator after the fourth hexadecimal digit: 00000308 has the same meaning as 0000-0308. Since in almost all cases at present the leading sixteen bits are zero, however, by convention the leading four hexadecimal zeros may be dropped entirely: the value 0308 is identical in meaning to the value 0000-0308.

In some cases, the character is represented not as a single UCS character but as a sequence of such characters; in this case, each thirty-two-bit value except the last must be followed by a plus sign:

```
<character class='lexical'>
  <form string='*=+U'
        entityStd="Ucdgr"
        ucs-4='03A5+0302+0308'
        afiicode='F300EC'>
            <desc>Greek capital letter Upsilon with
                  circumflex and diaresis</desc>
  </form>
</character>
```

If a given <character> element has more than one encoding using ISO 10646 (e.g. both as "a-umlaut" and as "a" plus "umlaut"), then both encodings may be given, separated by blanks:

```
<character class='lexical'>
  <form string=''
        entityStd="Auml"
        afiicode='F127"
        ucs-4='0041+0308 00C4' >
            <desc>Latin capital letter A with umlaut</desc>
  </form>
</character>
```

In most cases, identifying the character or character form by means of its UCS-4 and AFII codes will suffice to identify the character for all later users of the WSD. In some cases, however, further information must be provided. This may be provided in a <note> attached to the <character> or <form> element:

```
<character class='lexical'>
  <form string='N'
        entityLoc="nn" >
        ucs-4="0274"   >
            <desc>Standard ms symbol for double n.</desc>
  </form>
  <note>This character has the form of a capital-letter N,
        but is written the same height as a lower-case N.
        Its appearance is thus that of UCS-4 0274, but it
        does not have the same semantics.
  </note>
</character>
```

In some cases, it will be necessary or useful to provide an image of the character in question, or to refer to a standard reference work for such an image. The following <character> element might be used to describe, for example, a common Old French abbreviation for "est", for which the local entity *est* has been defined:

```
<character class='lexical'>
  <form string='' entityLoc="est">
     <desc>Old French abbreviation for 'est':  lowercase
           'e' with a tilde or macron above.</desc>
     <note>For an image of this character, see
           Cappelli, p. 113, column 1, line 4
           (leftmost and rightmost item).</note>
  </form>
</character>
```

Here, "Cappelli" is the name of a standard reference work which may be consulted to see what the character in question looks like.[1]

Where recourse to reference works is impossible, a picture of the character may be encoded using any standard graphics format, and associated with the character by standard SGML techniques. The SGML document must then have:

[1] *Dizionario di Abbreviature latine ed italiane* per cura di Adriano Cappelli, 6th ed. (Milan: Ulrico Hoepli, 1979). This work on Latin abbreviations might be less convenient for the purpose than one concentrating on Old French, but it is more widely used than any other.

- an SGML notation declaration for the graphics format used
- an external entity declaration for the file containing the image
- an <extFigure> element to name the notation and the entity

For a discussion of graphic images and of the declaration of non-SGML notations, see chapter 22 ('Tables, Formulae, and Graphics') on p. 523. If the Old French abbreviation is encoded using CGM (Computer Graphics Metafile) format in a file called *est.cgm*, then it may be associated with the appropriate character declaration as follows. In the DTD subset of the WSD, the following declarations are required:

```
<!NOTATION cgm PUBLIC 'ISO 8632/2//NOTATION
                       Computer Graphics Metafile
                       Character encoding//EN'>
<!ENTITY estFigure system 'est.cgm' NDATA cgm>
```

In the body of the WSD itself:

```
<character class='lexical'>
  <form string=''
        entityLoc="est">
            <desc>Old French abbreviation for 'est':  lowercase
                  'e' with a tilde or macron above.
            </desc>
            <extFigure notation='cgm' entity='estFigure'>
            <note>For an image of this character, see
                  Cappelli, p. 110, column 1, line 4
                  (leftmost and rightmost item).
            </note>
  </form>
</character>
```

Despite now having a picture of the character, we retain the prose description and reference to Cappelli, for the sake of those without ready access to the appropriate graphics processors.

The <exceptions> element and its contents are declared thus:

```
<!-- 25.4.2:  Exceptions to the base components               -->
<!ELEMENT exceptions    - O  (character*)                       >
<!ATTLIST exceptions         %a.global;                         >
<!ELEMENT character     - O  (desc*, form+, note*)              >
<!ATTLIST character          %a.global;
          class              (lexical | punc | lexpunc | digit
                             | space | DL | LD | dia | joiner |
                             other)              lexical        >
<!ELEMENT desc          - O  (#PCDATA)                          >
<!ATTLIST desc               %a.global;                         >
<!ELEMENT form          - O  (desc+, (figure | extFigure)*,
                             note*)                             >
<!ATTLIST form               %a.global;
          string             CDATA               #IMPLIED
          codedCharSet       IDREF               #IMPLIED
          entityStd          ENTITIES            #IMPLIED
          entityLoc          ENTITIES            #IMPLIED
          ucs-4              CDATA               #IMPLIED
          afiicode           CDATA               #IMPLIED       >
<!ELEMENT figure        - -  CDATA                              >
<!ATTLIST figure             %a.global;
          notation           NAME                #REQUIRED      >
<!ELEMENT extFigure     - O  EMPTY                              >
<!ATTLIST extFigure          %a.global;
          notation           NAME                #REQUIRED
          entity             ENTITY              #REQUIRED      >
<!-- This fragment is used in sec. 25.1                         -->
```

25.4.3 Documenting Coded Character Sets and Entity Sets

Public or private coded character sets and entity sets may be usefully documented using WSDs; the WSD will make explicit some information (such as the UCS-4 and AFII codes) not normally given explicitly in character set standards or public entity sets. The coded character set or entity set being documented should be included by means of a <codedCharSet> or <entitySet> element; the <exceptions> element should include one <character> element for each character included in the character set or the entity set. Deciding whether to treat two entities or two bit patterns as separate characters or as forms of the same character will require knowledge of the script involved, and different encoders may reach different decisions. In cases of doubt, though, it is usually acceptable practice to treat each bit pattern in a coded character set, and each entity in an entity set, as a distinct character.

A non-standard local coded character set (e.g. an EBCDIC character set) may be documented in a WSD by defining one <character> element for each printable code point in the character set, adding the names of standard (and local) entities, UCS-4 codes, and AFII codes as appropriate. Since this extra information is useful in packing documents for interchange, and in processing pattern arguments in the TEI extended-pointer syntax described in section 14.2 ('Extended Pointers') on p. 340, those responsible for a local installation are strongly encouraged to document the local system character set in a WSD, if it is not already so documented.

25.4.4 Documenting Transliteration Schemes

When a script is encoded not in a character set designed for it, but in one designed for another script, (e.g. Greek encoded using the Latin alphabet), a transliteration scheme is necessary. In documenting such a transliteration scheme, the coded character set actually in use should be named as a base component. An <exceptions> element can then be used to override the normal meaning of the individual byte strings used in the transliteration. For example, the following <character> element overrides the usual association of the byte representing A with the Latin letter A and substitutes instead an association with the Greek letter alpha:

```
<character class=lexical>
  <form string='A' entityStd="agr" ucs-4='03B1' afiicode='260061'>
    <desc>Greek small letter alpha</desc>
  </form>
</character>
```

Care should be taken in choosing or developing transliteration schemes to ensure that they are unambiguously reversible.

25.5 Notes in the WSD

Notes on the WSD, individual characters, or individual character forms may be included in the <note> element at the appropriate level.

<note> (in a writing system) contains a note of any type.

Unlike its counterpart in the main TEI DTD, the <note> element within the writing system declaration may contain no paragraphs and no phrase-level elements: only character data. It is formally declared thus:

```
<!-- 25.5:  Notes                                          -->
<!ELEMENT note          - O  (#PCDATA)                      >
<!ATTLIST note              %a.global;                      >
<!-- This fragment is used in sec. 25.1                    -->
```

25.6 Linkage between WSD and Main Document

The writing system declaration is associated with different portions of a main document by means of the global lang attribute. This attribute is defined as an SGML IDREF and its value must be the SGML identifier on a <language> element within the TEI header of the main document. The <language> element in turn provides, in its wsd attribute, the SGML name of the entity (usually an external file) containing the writing system declaration associated with that lang value. For a more detailed account of this process, compare the discussion in section 26.1 ('Linking a TEI Text to Feature System Declarations') on p. 589.

At least one writing system declaration must be associated with any TEI document: this follows from the requirement that a value be specified for the lang attribute on the outermost element (<tei.2> or <teiCorpus.2>) of any TEI document, since the lang attribute is required to point at a <language> element in the TEI header, which in turn is required to indicate an entity containing the writing system declaration associated with that language.

25.7 Predefined TEI WSDs

The Text Encoding Initiative has defined standard writing system declarations for the following languages and scripts:

- The nine official languages of the European Community (Danish, Dutch, English, French, Greek, German, Italian, Portuguese, and Spanish), encoded using the character set ISO 8859-1
- Hebrew (using ISO 8859-8)
- Russian (using ISO 8859-5)
- classical Greek (using the Thesaurus Linguae Graecae's Beta Code transliteration scheme)

Work is underway on a standard writing system declarations for Japanese; resources permitting, declarations for Korean and Chinese will also be made.

In addition to the language-specific WSDs just mentioned, the TEI has defined a number of WSDs to document specific public coded character sets and entity sets. These are used as components in the language-specific WSDs, and may be similarly used for locally developed WSDs. WSDs for specific ISO standard character sets include:

- ISO 646 (non-national subset)
- ISO 646 (International Reference Version)
- ISO 8859-1 (Latin 1, Western Europe)
- ISO 8859-2 (Latin 2, Eastern Europe)
- ISO 8859-5 (Latin and Cyrillic)
- ISO 8859-7 (Latin and Greek)
- ISO 8859-8 (Latin and Hebrew)
- ISO 8859-9 (Western Europe and Turkey)

Writing system declarations are also under preparation for a number of commercial character sets in wide use:

- IBM Code Page 437 (IBM PC, early models)
- IBM Code Page 850 (IBM PS/2 and later-model IBM PCs)
- IBM Code Page 1014
- Apple Macintosh default system character set
- Adobe Postscript default character set

The Text Encoding Initiative has prepared entity sets for use in transcribing some languages; these are distributed both as SGML entity sets and as TEI writing system declarations documenting those entity sets:

- -//TEI P3: 1994//ENTITIES Arabic//EN
- -//TEI P3: 1994//ENTITIES Coptic//EN
- -//TEI P3: 1994//ENTITIES Classical Greek//EN

- -//TEI P3: 1994//ENTITIES International Phonetic Alphabet//EN

WSDs will also be prepared for other standard entity sets, including:

- ISO 8879: 1986//ENTITIES Added Latin 1//EN
- ISO 8879: 1986//ENTITIES Added Latin 2//EN
- ISO 8879: 1986//ENTITIES Russian Cyrillic/EN
- ISO 8879: 1986//ENTITIES Non-Russian Cyrillic//EN
- ISO 8879: 1986//ENTITIES Greek Letters//EN
- ISO 8879: 1986//ENTITIES Diacritical Marks//EN
- ISO 8879: 1986//ENTITIES Box and Line Drawing//EN
- ISO 8879: 1986//ENTITIES Numeric and Special Graphic//EN
- ISO 8879: 1986//ENTITIES Publishing//EN
- ISO 8879: 1986//ENTITIES General Technical//EN

All TEI writing system declarations are distributed with the TEI document type definitions.

The standard TEI writing system declarations are expected to meet the needs of many encoders; some, however, will need to prepare new WSDs to describe character-encoding schemes not included in the standard WSDs.

25.8 Details of WSD Semantics

This section describes the meaning of the WSD in more formal terms than have been used elsewhere in this chapter; it can be skipped by most readers, but should be read carefully by those who wish to write complex writing system declarations or to implement software to process writing system declarations or to interpret them in the processing of TEI-conformant documents.

25.8.1 WSD Semantics: General Principles

A writing system declaration provides a complicated bundle of mappings:

- a 1:1 partial function from strings in given coded character sets to character forms
- a function from entity names to character forms, and therefore derivatively a function from entity names to strings
- a function from character forms to characters, and therefore derivatively:
  - a function from strings to characters
  - a function from entity names to characters
- a relation between UCS-4 codes and character forms
- a relation between AFII codes and character forms
- a function from UCS-4 codes to characters
- a relation from AFII codes to characters

To ensure that the relations described as functions are in fact functional, the following constraints apply on the WSD:

- No two <form> elements can have the same values for both codedCharSet and string. Since usually there is only one <codedCharSet> used as a basic component, this usually means each string attribute value must be unique in the WSD.
- No two <form> elements can name the same entity in either entityStd or entityLoc. It is legal, though pointless, for both entityStd and entityLoc on the same <form> element to name the same entity.
- More than one <form> element may have the same UCS-4 value, but if so they must be within the same <character> element.

These constraints may be summarized thus: one "character" (however the creator of the WSD defines a character) can be associated with more than one byte string, entity name, UCS-4 code, or AFII code, but any single byte string (given a specific coded character set), any single entity name, and any single UCS-4 code must be associated with only one single <character>. One can, for example, associate both "tilde" and "logical not" with a <character> meaning

"logical negation", but one cannot associate both a <character> called "tilde" and one called "logical negation" with the ASCII character 7/14: given a 7/14 in the text, it must be unambiguously clear whether the character is a "tilde" or a "logical negation". If one wishes to retain the ambiguity, one must define a <character> called (for example) "logical-not or tilde or swung-dash". Similar restrictions apply to entity names and UCS-4 codes: each must be associated with a single <character> element.

25.8.2 Semantics of WSD Base Components

The effects of naming coded character sets, entity sets, and other WSDs as base components may now be defined thus:

- reference to a coded character set makes available the set of bit-pattern-to-character mappings defined in the coded character set. That is, if a WSD refers to a coded character set, then whenever the WSD is in use, any character in that coded character set may be used with its standard meaning unless it has been redefined using the <exceptions> element. It is recommended that a WSD be provided for each coded character set, to make the mappings fully explicit.
- reference to an entity set makes available the set of entity-name-to-character mappings defined in the entity set. It is assumed that standard public entity sets contain enough information to count as a valid mapping; for private entity sets, the preferred method of providing the necessary information is to define the entity set in a WSD. If for example a WSD refers to the ISO Latin 1 entity set, then whenever that WSD is in use, any entity in that set may be used with its public meaning, unless it has been redefined in the <exceptions> element.
- reference to a WSD makes available the set of mappings declared in that WSD; the language and writing system direction information given in the base WSD is ignored.

If reference is made only to standard character sets and entity sets, there is no mechanical method of associating the "characters" involved in one mapping with those involved in another. E.g. a reference to ISO 646 IRV provides a map from code point 5/11 to a character one might call "left square bracket". A reference to entity set *ISOpub1* provides a map from the entity name *lbr* to what should probably be considered the same character. There is however no guarantee that any processing software will necessarily be sufficiently intelligent to make this association of mappings automatically; it requires hard-coded knowledge of the specifics of certain character sets and entity sets.

When, however, base WSDs are used to document important entity sets and character sets, it does become possible to define mechanical methods of associating <character> elements in different base components.

25.8.3 Multiple Base Components

When multiple bases of the same type are referred to, the effects are these:

- if more than one coded character set is named, then it is expected that character-set shifting as described in ISO 2022 or some equivalent is in use, and proper shifting is the responsibility of the user. All strings in the WSD must specify the ID of the proper coded-character-set base, using the codedCharSet attribute.
- if more than one entity set is named, then entity names from all named sets may be used as values of the entityStd and entityLoc attributes. If the same name occurs in more than one entity set, the assumption is made that it refers each time to the same character.
- if more than one base WSD is named, then all characters declared in all the WSDs are available. For this case, we can define what happens to merge the different base components more precisely than for the other types of base component:

  - any two <form> elements which name the same entity or the same string in the same coded character set are considered the same form, and are merged as described below in section 25.8.5 ('Merger of Form and Character Elements') on p. 587.

– any two <form> elements which give the same UCS-4 code are considered forms of the same <character>, and their parent <character> elements are merged. The forms themselves may be merged or may remain distinct: if the forms have conflicting values for any attribute, they must remain distinct; if they don't conflict, they may be merged, at the option of the processing software. In the general case, there might be more than one way to perform mergers, so merger is not required.

The result of invoking multiple base WSDs is thus a merged WSD in which the <form> and <character> elements have been merged as prescribed. If the merger is impossible because the two WSDs are incompatible, a semantic error occurs. A set of WSDs is compatible and may be invoked together if all of the following are true:

- any given entity name is associated with a single string (in a given coded character set) and a single character class
- any given string or UCS-4 code is associated with a single character class

25.8.4 Semantics of Exceptions

We can now define the semantics of the <exceptions> element.

The base components provide a preliminary set of mappings, as described above. For convenience let us call this the *default map*. The <exceptions> element allows the user to modify the default map by defining further mappings and by overriding parts of the default map. There are three cases: a new <character> element replaces an old one, is merged with an old one, or is added to the set without affecting any old ones.

25.8.4.1 Case 1: replacement

If a <form> element within <exceptions> (F-new) "collides" with a <form> element in the default map (F-old), then the parent <character> element of F-new replaces the parent element of F-old. Two <form> elements collide if they have the same values for codedCharSet and string. (N.B. if this condition occurs within the default map, the two <form> elements are merged.) For example, to define the TLG Beta code transliteration of alpha as a we first name ISO 646 IRV as a base component; this has the effect of creating the following (possibly imaginary) <form> element:

```
<character id=a class=lexical>
  <form string='a'>
    <desc>lowercase latin letter A</desc>
  </form>
</character>
```

We then include the following within the <exception> element:

```
<character id=alpha class=lexical>
  <form string='a' entityStd='gkalpha'>
    <desc>lowercase Greek alpha</desc>
  </form>
</character>
```

This overrides the <character> element for latin A, and indicates that in the transliteration scheme documented by this WSD, character 6/01 represents a Greek alpha, no matter what ISO 646 says.

25.8.4.2 Case 2: merger

If a <character> element within <exceptions> "overlaps" with one in the default map, then the two <character> elements are merged. Two <character> elements overlap if any of their <form> elements name the same entity or UCS-4 code. (N.B. if these conditions occur within the default map, they lead to merger either of the two <form> elements — for entity name overlap — or of the two <character> elements.) For example: suppose we wish to document the three-Rs transcription described in section 4.1.2 ('Characters Not Available Locally') on p. 72. We name ISO 646 IRV as a base character set (or WSD) and add the following exceptions:

```
<exceptions>
  <character id=r class=lexical>
      <desc>lowercase latin letter r</desc>
      <form string='' entityLoc='r' ucs-4='0072'>
          <desc>'normal' form, similar to modern print r and to
                Cappelli, p. 318, line 2, items 3, 6, 15.</desc>
      </form>
      <form string='' entityLoc='r2' ucs-4='0072'>
          <desc>'round' form, usually following 'o', similar
                to a modern Arabic digit 2 (or to Cappelli, p. 318,
                line 2, items 13 and 14)</desc>
      </form>
      <form string='' entityLoc='r3' ucs-4='0072'>
          <desc>'small-cap' form, like a capital R but
                same height as lowercase (cf. Cappelli, p. 318,
                line 1, items 2 and 3)</desc>
      </form>
  </character>
</exceptions>
```

As a second example, imagine we wish to document a local entity set for Old English in which we define local entities t (thorn), d (eth), and a (aesc). Assuming the TEI has provided a WSD for the Latin 1 entities, the whole WSD is this:

```
<writingSystemDeclaration
       name='-//OTA 1990//NOTATION WSD Old English entities//EN'
       date='1993-05-25'
       lang=eng>
  <language iso639=''>Various</language>
  <script>Latin alphabet, extended</script>
  <direction lines=TB chars=LR>
  <characters>
       <baseWsd name='-//TEI P3: 1994//NOTATION
                      WSD ISO Added Latin 1//EN'
                authority=TEI>
       <exceptions>
           <character class=lexical>
               <form entityStd='thorn' entityLoc='t'>
                   <desc>lowercase latin letter thorn</desc>
                   </form></character>
           <character class=lexical>
               <form entityStd='Thorn' entityLoc='T'>
                   <desc>uppercase latin letter thorn</desc>
                   </form></character>
           <character class=lexical>
               <form entityStd='eth' entityLoc='d'>
                   <desc>lowercase latin letter eth</desc>
                   </form></character>
           <character class=lexical>
               <form entityStd='Eth' entityLoc='D'>
                   <desc>uppercase latin letter eth</desc>
                   </form></character>
           <character class=lexical>
               <form entityStd='aelig' entityLoc='a'>
                   <desc>lowercase latin aesc (= digraph aelig)</desc>
                   </form></character>
           <character class=lexical>
               <form entityStd='AElig' entityLoc='A'>
                   <desc>uppercase latin aesc (= digraph aelig)</desc>
                   </form></character>
       </exceptions>
  </characters>
```

```
  <note>This WSD is just to document the local entities; it should be
        named as a base WSD by the actual writing system declaration.
  </note>
</writingSystemDeclaration>
```

This has the effect of merging the <character> elements for thorn, eth, and aesc (or a-e ligature) defined in the ISO Latin 1 WSD with those given here, which specify the local entity name. The <form> elements may or may not be merged, so the software may or may not actually realize that the local entity *t* corresponds with the UCS-4 code given in the TEI WSD for ISO Latin 1.

The full local WSD can then be this:

```
<writingSystemDeclaration
       name='-//OTA 1993//NOTATION Old English WSD//EN'
       date='1993-05-25'
       lang=eng>
  <language iso639=ang>Anglo-Saxon / Old English</language>
  <script>Latin alphabet, extended</script>
  <direction lines=TB chars=LR>
  <characters>
        <baseWsd name='-//TEI P3: 1994//NOTATION WSD ISO 646 IRV//EN'
                 authority='TEI'>
        <baseWsd name='-//OTA 1990//NOTATION
                       WSD Old English entities//EN'
                 authority='private'>
        <baseWsd name='-//TEI P3: 1994//NOTATION
                       WSD ISO Added Latin 1//EN'
                 authority='TEI'>
  </characters>
</writingSystemDeclaration>
```

We refer explicitly to ISO Latin 1, for clarity, but in theory it has already been included in *-//OTA 1990//WSD Old English entities//EN* and need not be repeated. At this time, the rules for merger would force our local <form> elements to be merged with the standard <form> elements, so the local entity *t* would map correctly into the UCS-4 character set.

25.8.4.3 Case 3: expansion

If a <character> element has no <form> children which collide with anything in the default map, and does not itself overlap with anything in the default map, then it is simply added to the default map. For example, suppose we wish to document an abbreviation used for Old French "est" in our manuscript, which resembles e with a tilde or macron. Since we expect we may have more abbreviations for "est", we use the local entity name *est1* for this one. Within <exceptions>, we declare the abbreviation thus:

```
<character id=est1 class=lexical>
  <form string='' entityLoc='est1'>
    <desc>abbreviation for 'est', lowercase latin e
          with a tilde or macron above, similar to
          Cappelli p. 113, col 1, items 4(a) and 8.</desc>
  </form>
</character>
```

25.8.5 Merger of Form and Character Elements

In some cases, the <form> and <character> elements introduced notionally by reference to a coded character set or entity set, or introduced explicitly by reference to a base WSD, may be considered as referring to identical objects; this is called *merger*. Two <form> elements F1 and F2 can be merged if they both have the same values for codedCharSet and string, or if codedCharSet and string are unspecified (implied) in at least one. When F1 and F2 are merged, the result is a (possibly imaginary) <form> element (F3) the attributes of which are derived thus:

- if F1 has no value for a codedCharSet, then F3 has the same value for this attribute as does F2. If both F1 and F2 have explicit values, the values must be identical.
- if F1 has an empty string for string, then F3 has the same value as F2. If both have values other than the empty string, they must be identical.
- for entityStd, entityLoc, ucs-4, and afiiCode, F3 gets a value containing all the entity names or codes which appear in the corresponding attribute values of either F1 or F2. I.e. the attribute values are viewed as sets, and F3 gets the union of F1 and F2.

The children of the new element are derived by taking all the <desc> children of F1, then all the <desc> children of F2; all the <figure> children of F1, then those of F2; all the <note> children of F1, then all the <note> children of F2. In other words, all the children of the source elements survive as children of the result element.

Provided that their forms are compatible, two <character> elements C1 and C2 may be merged unless their values for class differ. The resulting <character> element C3 has the same value for its class attribute as C1 and C2, and all the children of C1 and C2 are made children of C3 (<desc> children first, then <form> children).

Note that merger is sometimes required by the semantic rules given above, and sometimes optional. If merger is required but not legal (because the two elements to be merged are incompatible), then a semantic error has occurred and the two base WSDs which give rise to it should not be invoked together.

Chapter 26

Feature System Declaration

The Feature System Declaration (FSD) is an auxiliary file used in conjunction with a TEI-conforming text that makes use of <fs> (that is, feature structure) elements. The FSD serves three purposes:

- It provides a mechanism by which the encoder can list all of the feature names and feature values and give a prose description as to what each represents.
- It provides a mechanism by which the encoder can define constraints on what it means to be a well-formed feature structure. These constraints may involve constraints on the range of a feature value, constraints on what features are valid within certain types of feature structures, or constraints that prevent the co-occurrence of certain feature-value pairs.
- It provides a mechanism by which the encoder can define the intended interpretation of underspecified feature structures. This involves defining default values (whether literal or computed) for missing features.

As a component of the interchange standard for encoded text, the FSD serves an important function in documenting precisely what the encoder intended by the system of feature structure markup used in the encoded text. As application software is developed which makes use of encoded texts, the FSD will also become an important resource that will allow software to validate the feature structure markup in a text and to infer the full interpretation of underspecified feature structures.

This chapter begins by describing how the encoded text uses header information to make links to any associated FSDs. The second through fourth sections describe the overall structure of an FSD and give details of how to encode its parts. The final section offers a full example. [1]

26.1 Linking a TEI Text to Feature System Declarations

In order for application software to use feature system declarations to aid in the automatic interpretation of encoded texts, or even for human readers to find the appropriate declarations which document the feature system used in markup, there must be a formal link from the encoded texts to the declarations. As it turns out, the mechanism for linking texts to FSDs parallels the mechanism for linking texts to writing system declarations (WSDs).

The linkage is made in two places. First, in the DTD subset of the document type declaration at the beginning of the text file, an external entity is defined for each FSD that is associated with the encoded text. That entity declaration associates an entity name with the name of a file on the host system. It appends the SUBDOC keyword to tell the processor that the named file is a self-contained SGML document. See the example below for details of syntax.

[1] For a fuller discussion of the reasoning behind FSDs and for another complete example, see "A rationale for the TEI recommendations for feature-structure markup," by D. Terence Langendoen and Gary F. Simons (to appear in the special TEI issue of *Computers and the Humanities*).

The second place in which the linkage from text to FSDs is made is in the TEI header, as mentioned in section 5.3.7 ('The Feature System Declaration') on p. 105. Within the <encodingDesc> element, a special <fsdDecl> element may be used for each distinct feature structure type, as follows:

<fsdDecl> identifies the feature system declaration which contains definitions for a particular type of feature structure. Attributes include:

- **type** identifies the type of feature structure documented in the FSD; this will be the value of the type attribute on at least one feature structure.
- **fsd** specifies the external entity containing the feature system declaration; an entity declaration in the document's DTD subset must associate the entity name with a file on the system.

Note that one <fsdDecl> element must be specified for each distinct type of feature structure used in the markup. The fsd element supplies the name of the external entity containing the actual declaration for that type of feature structure.

There may be multiple <fsdDecl> elements for a given FSD; one for each type of feature structure it defines. For instance, in the following example, the file *lex.fsd* contains an FSD that contains definitions of feature structures for both lexical entries (<fs type=entry>) and lexical subentries (<fs type=subentry>).

The following example shows the markup for linking a TEI document to two WSDs and two FSDs. The linkage to both WSDs and FSDs is shown in order to illustrate the parallel nature of the linking mechanisms for both kinds of auxiliary files.

```
<!DOCTYPE tei.2 system 'tei2.dtd' [
   <!-- In the DTD subset, we declare external
        entities for our FSDs and WSDs.        -->
   <!ENTITY wsd.english system 'en.wsd'   SUBDOC >
   <!ENTITY wsd.french  system 'fr.wsd'   SUBDOC >
   <!ENTITY fsd.gazdar  system 'gpsg.fsd' SUBDOC >
   <!ENTITY fsd.lexicon system 'lex.fsd'  SUBDOC >
]>
<tei.2>

<teiHeader>
   <!-- In the encoding description of the TEI header,
        we link the FSDs to the fs TYPE attribute;
        in the profile description,
        we link the WSDs to the global LANG attribute.
        -->
   <fileDesc> ... </fileDesc>
   <encodingDesc>
        <!-- ... -->
        <fsdDecl type=GPSG     fsd=fsd.gazdar>
        <fsdDecl type=entry    fsd=fsd.lexicon>
        <fsdDecl type=subentry fsd=fsd.lexicon>
        <!-- ... -->
   </encodingDesc>
   <profileDesc>
        <!-- ... -->
        <langUsage>
            <language id=EN wsd=wsd.english>English</>
            <language id=FR wsd=wsd.french >French</>
        </langUsage>
   </profileDesc>
</teiHeader>
<!-- The text goes here -->
</TEI.2>
```

The auxiliary tag set for feature system declarations is contained in the file *teifsd2.dtd*, which has the overall structure shown below:

```
<!-- 26.1:  Feature System Declaration                          -->
<!-- Text Encoding Initiative: Guidelines for Electronic        -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.      -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy      -->
<!-- in any form is granted, provided this notice is            -->
<!-- included in all copies.                                    -->

<!-- These materials may not be altered; modifications to       -->
<!-- these DTDs should be performed as specified in the         -->
<!-- Guidelines in chapter "Modifying the TEI DTD."             -->

<!-- These materials subject to revision. Current versions      -->
<!-- are available from the Text Encoding Initiative.           -->
<!-- First, we declare basic parameter entities and embed       -->
<!-- auxiliary files.                                           -->

<!-- Embed entities for TEI generic identifiers.                -->

<!ENTITY % TEI.elementNames system 'teigis2.ent'                 >
%TEI.elementNames;

<!-- Embed entities for TEI keywords.                           -->

<!ENTITY % TEI.keywords.ent system 'teikey2.ent'                 >
%TEI.keywords.ent;

<!-- Define element classes for content models, shared          -->
<!-- attributes for element classes, and global attributes.     -->
<!-- (This all happens within the file teiclas2.ent.)           -->

<!ENTITY % TEI.elementClasses system 'teiclas2.ent'              >
%TEI.elementClasses;

<!-- Define element classes for feature structure               -->
<!-- declarations.                                              -->

<!ENTITY % x.boolean ''                                          >
<!ENTITY % m.boolean '%x.boolean any | none'                     >
<!ENTITY % x.binary ''                                           >
<!ENTITY % m.binary '%x.binary minus | plus'                     >
<!ENTITY % x.singleVal ''                                        >
<!ENTITY % m.singleVal '%x.singleVal %m.binary; | %m.boolean; |
           dft | msr | nbr | rate | str | sym | uncertain'       >
<!ENTITY % x.complexVal ''                                       >
<!ENTITY % m.complexVal '%x.complexVal alt | fs | vAlt'          >
<!ENTITY % x.featureVal ''                                       >
<!ENTITY % m.featureVal '%x.featureVal %m.complexVal; |
           %m.singleVal; | null'                                 >

<!-- Now, we declare the elements for FSDs proper.              -->

<!-- ... declarations from section 26.2                         -->
<!--     (Feature System Declaration)                           -->
<!--     go here ...                                            -->
<!-- ... declarations from section 26.3                         -->
<!--     (Feature definitions)                                  -->
<!--     go here ...                                            -->
<!-- ... declarations from section 26.4                         -->
<!--     (Feature structure constraints)                        -->
```

```
<!--     go here ...                                          -->

<!-- The elements for feature structures themselves are       -->
<!-- declared in teifs2.dtd                                   -->

<!ENTITY % TEI.fs.dtd system 'teifs2.dtd'                      >
%TEI.fs.dtd;

<!-- Finally, embed the TEI header and core tag sets.         -->

<!ENTITY % TEI.header.dtd system 'teihdr2.dtd'                 >
%TEI.header.dtd;
<!ENTITY % TEI.core.dtd system 'teicore2.dtd'                  >
%TEI.core.dtd;
```

26.2 The Overall Structure of a Feature System Declaration

A feature system declaration is encoded as a document of type <teiFsd2>. It has two parts: an obligatory header (which provides bibliographic information for the file) and a set of feature structure declarations (each of which defines one type of feature structure). Each feature structure declaration in turn has three parts: an optional description (which gives a prose comment on what that type of feature structure encodes), an obligatory set of feature declarations (which specify range constraints and default values for the features in that type of structure), and optional feature structure constraints (which specify co-occurrence restrictions on feature values). The header is encoded as a <teiHeader>, just as for any TEI.2 document; see chapter 5 ('The TEI Header') on p. 77. The other components listed above are unique to feature system declarations. Thus, the following new elements are involved:

<teiFsd2> contains a feature system declaration.

<fsDecl> declares one type of feature structure. Attributes include:

- **type** gives a name for the type of feature structure being declared.
- **baseType** gives the name of the feature structure type from which this type inherits features and constraints; if this type declares a feature with the same name as a feature of the base type, the definition within this <fsDecl> overrides the inherited definition. The <fsConstraints> are inherited only if this <fsDecl> does not specify any; otherwise the constraints in this <fsDecl> override. When no baseType is specified, no features or constraints are inherited.

<fsDescr> describes in prose what is represented by the type of feature structure declared in the enclosing <fsDecl>.

<fDecl> declares a single feature, specifying its name, organization, range of allowed values, and optionally its default value.

<fsConstraints> specifies constraints on the content of well formed feature structures.

Feature declarations and feature structure constraints are described in the next two sections of this chapter. Note that the specification of similar <fsDecl> elements can be simplified by devising an inheritance hierarchy for the feature structure types. Each <fsDecl> may name a baseType from which it inherits feature declarations and constraints. For instance, suppose that <fsDecl type=Basic> contains <fDecl name=One> and <fDecl name=Two>, and that <fsDecl type=Derived baseType=Basic> contains just <fDecl name=Three>. Then any instance of <fs type=Derived> may include all three features. This is because <fsDecl type=Derived> inherits the two feature declarations from <fsDecl type=Basic> when it specifies a baseType of Basic.

The following sample shows the overall structure of a complete FSD. Note that as a stand-alone document it begins with a DOCTYPE declaration which identifies the associated DTD.

```
<!DOCTYPE teiFsd2 system 'teifsd2.dtd'>
<teiFsd2>
   <teiHeader>
      <!-- The header is as for any TEI.2 document -->
   </teiHeader>
   <fsDecl type=SomeName>
      <fsDescr>Describes what this type of fs represents</>
      <fDecl name=featureOne>
         <!-- The declaration for featureOne -->
      </fDecl>
      <fDecl name=featureTwo>
         <!-- The declaration for featureTwo -->
      </fDecl>
      <fsConstraints>
         <!-- The feature structure constraints go here -->
      </fsConstraints>
   </fsDecl>
   <fsDecl type=AnotherType>
         <!-- Declare another type of feature structure -->
   </fsDecl>
</teiFsd2>
```

The formal definition of <teiFsd2> and feature structure declarations is as follows:

```
<!-- 26.2:  Feature System Declaration                        -->
<!ELEMENT teiFsd2       - -  (teiHeader, fsDecl+)              >
<!ATTLIST teiFsd2            %a.global;                        >
<!ELEMENT fsDecl        - -  (fsDescr?, fDecl+, fsConstraints?)
                                                               >
<!ATTLIST fsDecl             %a.global;
          type               CDATA               #REQUIRED
          baseType           CDATA               #IMPLIED      >
<!ELEMENT fsDescr       - O  (%paraContent;)                   >
<!ATTLIST fsDescr            %a.global;                        >
<!-- This fragment is used in sec. 26.1                       -->
```

26.3 Feature Declarations

Each feature is declared in an <fDecl> element whose name attribute identifies the feature being declared; this matches the name attribute of the <f> elements it declares. An <fDecl> also has an org attribute which declares the organizing principle for the values of the <f> elements it declares. That is, the value may be a unit (a single value), a set (in which the order is not significant and there are no duplicates), a bag (in which the order is not significant but duplicates are allowed), or a list (in which the order is significant). (See definition of org attribute of <f> in section 16.6 ('Singleton, Set, Bag and List Collections of Values') on p. 410.) An <fDecl> has three parts: an optional prose description (which should explain what the feature and its values represent), an obligatory range specification (which declares what values the feature is allowed to have), and an optional default specification (which declares what default value should be supplied when the named feature does not appear in an <fs>). A single unconditional default value may be specified, or multiple conditional values. If no default is specified, or if none of the conditions is met, then the default value is <none>; in other words, the feature is not applicable (see section 16.8 ('Boolean, Default and Uncertain Values') on p. 417 for a discussion of the <none> element).

The tags used in feature declarations are the following:

<fDecl> declares a single feature, specifying its name, organization, range of allowed values, and optionally its default value. Attributes include:

- **name** indicates the name of the feature being declared; matches the name attribute of <f> elements in the text.
- **org** specifies the organizing discipline of the feature value. Legal values are:

unit unitary atomic value

set set value (unordered, no duplicates)

bag bag value (unordered, may have duplicates)

list list value (ordered, may have duplicates)

<fDescr> describes in prose what is represented by the feature being declared and its values.

<vRange> defines the range of allowed values for a feature, in the form of an <fs>, <vAlt>, or primitive value; for the value of an <f> to be valid, it must be *subsumed* by the specified range; if the <f> contains multiple values (as sanctioned by the org attribute), then each value must be subsumed by the <vRange>.

<vDefault> declares the default value to be supplied when a feature structure does not contain an instance of <f> for this name; if unconditional, it is specified as one (or, depending on the value of the org attribute of the enclosing <fDecl>) more <fs> elements or primitive values; if conditional, it is specified as one or more <if> elements; if no default is specified, or no condition matches, the value <none> is assumed.

<if> defines a conditional default value for a feature; the condition is specified as a feature structure, and is met if it *subsumes* the feature structure in the text for which a default value is sought.

<then> separates the condition from the default in an <if>, or the antecedent and the consequent in a <cond> element.

The logic for validating feature values and for matching the conditions for supplying default values is based on the operation of subsumption. Subsumption is a standard operation in feature-structure-based formalisms. Informally, a feature structure *fs* subsumes all feature structures that are at least as informative as itself; that is, all feature structures that specify at least as many features as *fs* with values at least as informative as those given in *fs* (Pereira 1987:6; see also Shieber 1986:14-16). [2] A more formal definition requires that we first define the notion of "domain of a feature structure." A feature structure can be viewed as a partial function that maps features onto values; when viewed in this way, the domain of a feature structure is the set of top-level features it contains (that is, excluding features in embedded feature structures). We can now offer a more precise definition:

"*fs* subsumes *fs′* if both are identical primitive values, or if the domain of *fs* is a subset of the domain of *fs′*, and for every feature *f* in the domain of *fs*, the value of *f* in *fs* subsumes the value of *f* in *fs′*."

Following the spirit of the informal definition above, we can extend subsumption in a straightforward way to cover alternation, negation, special primitive values, and the use of attributes in the SGML markup. For instance, a <vAlt> containing the value v subsumes v. The negation REL=ne of value v subsumes any value that is not v. The value <unknown> subsumes any value. The value <any> subsumes any value that is in the range of a feature. <fs type=X></fs> subsumes any feature structure with TYPE=X. <nbr rel=ge value=0> subsumes any <nbr> with value greater than or equal to zero.

As an example of feature declarations, consider the following extract from *Generalized Phrase Structure Grammar* by Gerald Gazdar, Ewan Klein, Geoffrey Pullum, and Ivan Sag (Harvard University Press, 1985). In the appendix to their book (pages 245-247), they propose a feature system for English of which this is just a sampling:

```
feature     value range

INV         {+, -}
CONJ        {and, both, but, either, neither, nor, or, NIL}
COMP        {for, that, whether, if, NIL}
AGR         CAT
PFORM       {to, by, for, ...}

Feature specification defaults
```

[2] Fernando C. N. Pereira, *Grammars and logics of partial information*, SRI International Technical Note 420 (Menlo Park, CA: SRI International, 1987), and Stuart Shieber, *An Introduction to Unification-based Approaches to Grammar*, CSLI Lecture Notes 4 (Palo Alto, CA: Center for the Study of Language and Information, 1986).

```
FSD 1:  [-INV]
FSD 2:  ~[CONJ]
FSD 9:  [INF, +SUBJ] --> [COMP for]
```

The INV feature, which encodes whether or not a sentence is inverted, allows only the values plus (+) and minus (-). If the feature is not specified, then the default rule (FSD 1 above) says that a value of minus is always assumed. The feature declaration for this feature would be encoded as follows:

```
<fDecl name=INV>
   <fDescr>inverted sentence</fDescr>
   <vRange>
       <vAlt><plus><minus></vAlt></vRange>
   <vDefault>
       <minus></vDefault>
</fDecl>
```

The value range is specified as an alternation (more precisely, an exclusive disjunction) of <plus> and <minus>. That is, the value must be one or the other, but not both or neither.

The CONJ feature indicates the surface form of the conjunction used in a construction. The in the default rule (see FSD 2 above) represents negation. This means that by default the feature is not applicable, in other words, no conjunction is taking place. This corresponds to the simple value <none>; see section 16.8 ('Boolean, Default and Uncertain Values') on p. 417. Note that this is distinct from the NIL value allowed in the value range. In their analysis, NIL means that the phenomenon of conjunction is taking place but there is no explicit conjunction in the surface form of the sentence. The feature declaration for this feature would be encoded as follows:

```
<fDecl name=CONJ>
   <fDescr>surface form of the conjunction</fDescr>
   <vRange>
      <vAlt>
         <sym value=and>
         <sym value=both>
         <sym value=but>
         <sym value=either>
         <sym value=neither>
         <sym value=nor>
         <sym value=or>
         <sym value=NIL>
      </vAlt></vRange>
   <vDefault>
      <none></vDefault>
</fDecl>
```

Note that the <vDefault> is not strictly necessary in this case, since <none> is the value assumed in the absence of a default specification.

The COMP feature indicates the surface form of the complementizer used in a construction. In value range, it is analogous to CONJ. However, its default rule (see FSD 9 above) is conditional. It says that if the verb form is infinitival (the VFORM feature is not mentioned in the rule since it is the only feature that can take INF as a value), and the construction has a subject, then a 'for' complement must be used. For instance, to make John the subject of the infinitive in 'It is necessary to go,' a 'for' complement must be used; that is, 'It is necessary for John to go.' The feature declaration for this feature would be encoded as follows:

```
<fDecl name=COMP>
   <fDescr>surface form of the complementizer</fDescr>
   <vRange>
      <vAlt>
         <sym value=for>
         <sym value=that>
         <sym value=whether>
         <sym value=if>
         <sym value=NIL>
```

```
      </vAlt></vRange>
   <vDefault>
      <if><fs><f name=VFORM><sym value=INF></f>
              <f name=SUBJ><plus></f></fs>
      <then><sym value=for></if>
      </vDefault>
</fDecl>
```

The AGR feature stores the features relevant to subject-verb agreement. Gazdar et al. specify the range of this feature as CAT. This means that the value is a *category*, which is their term for a feature structure. This is actually too weak a statement. Not just any feature structure is allowable here; it must be a feature structure for agreement (which is defined in the complete example at the end of the chapter to contain the features of person and number). The following feature declaration encodes this constraint on the value range:

```
<fDecl name=AGR>
   <fDescr>agreement for person and number</fDescr>
   <vRange><fs type=Agreement></fs></vRange>
</fDecl>
```

That is, the value must be a feature structure of type `Agreement`. The complete example at the end of this chapter includes the `<fsDecl type=Agreement>` which includes `<fDecl name=PERS>` and `<fDecl name=NUM>`.

The PFORM feature indicates the surface form of the preposition used in a construction. Since PFORM is specified above as an open set, `<str>` is used in the range specification below rather than `<sym>`.

```
<fDecl name=PFORM>
   <fDescr>word form of a preposition</fDescr>
   <vRange><str rel=ne></str></vRange>
</fDecl>
```

This example makes use of a negation. `<str rel=ne></str>` subsumes any string that is not the empty string.

The formal definition for feature declarations follows. Note that the class *featureVal* includes all possible single feature values, including a `<vAlt>`.

```
<!-- 26.3:  Feature definitions                                  -->
<!ELEMENT fDecl         - -  (fDescr?, vRange, vDefault?)       >
<!ATTLIST fDecl              %a.global;
          name               NMTOKEN             #REQUIRED
          org                (unit | set | bag | list)
                                                 unit           >
<!ELEMENT fDescr        - O  (%paraContent;)                    >
<!ATTLIST fDescr             %a.global;                         >
<!ELEMENT vRange        - O  (%m.featureVal)                    >
<!ATTLIST vRange             %a.global;                         >
<!ELEMENT vDefault      - -  ((%m.featureVal)+ | if+)           >
<!ATTLIST vDefault           %a.global;                         >
<!ELEMENT if            - -  ((fs | f | fAlt), then,
                              (%m.featureVal) )                 >
<!ATTLIST if                 %a.global;                         >
<!ELEMENT then          - O  EMPTY                              >
<!ATTLIST then               %a.global;                         >
<!-- This fragment is used in sec. 26.1                         -->
```

26.4 Feature Structure Constraints

Ensuring the validity of feature structures may require much more than simply specifying the range of allowed values for each feature. There may be constraints on the co-occurrence of one

feature value with the value of another feature in the same feature structure or in an embedded feature structure.

Such constraints on valid feature structures are expressed as a series of conditional and biconditional tests in the <fsConstraints> part of an <fsDecl>. A particular feature structure is valid only if it meets all the constraints. The <cond> element encodes the conventional if-then conditional of boolean logic which succeeds when both the antecedent and consequent are true, or whenever the antecedent is false. The <bicond> element encodes the biconditional (if and only if) operation of boolean logic. It succeeds only when both antecedent and consequent are true, or both are false. In feature structure constraints the antecedent and consequent are expressed as feature structures; they are considered true if they *subsume* (see section 26.3 ('Feature Declarations') on p. 593) the target feature structure. The following elements make up the <fsConstraints> part of an FSD:

<fsConstraints> specifies constraints on the content of well formed feature structures.

<cond> defines a conditional feature-structure constraint; the consequent and the antecedent are specified as feature structures or feature-structure groups; the constraint is satisfied if both the antecedent and the consequent *subsume* a given feature structure, or if the antecedent does not.

<bicond> defines a biconditional feature-structure constraint; both consequent and antecedent are specified as feature structures or groups of feature structures; the constraint is satisfied if both *subsume* a given feature structure, or if both do not.

<then> separates the condition from the default in an <if>, or the antecedent and the consequent in a <cond> element.

<iff> separates the condition from the consequence in a <bicond> element.

For an example of feature structure constraints, consider the following "feature co-occurrence restrictions" extracted from the feature system for English proposed by Gazdar, Klein, Pullum, and Sag (1985:246-247):

FCR 1: [+INV] ⟶ [+AUX, FIN]
FCR 7: [BAR 0] ≡ [N] & [V] & [SUBCAT]
FCR 8: [BAR 1] ⟶ [SUBCAT]

The first constraint says that if a construction is inverted, it must also have an auxiliary and a finite verb form. That is,

```
<cond><fs><f name=INV><plus></f></fs>
      <then>
      <fs>
         <f name=AUX><plus></f>
         <f name=VFORM><sym value=FIN></f>
      </fs>
</cond>
```

The second constraint says that if a construction has a BAR value of zero (i.e., it is a sentence), then it must have a value for the features N, V, and SUBCAT. By the same token, because it is a biconditional, if it has values for N, V, and SUBCAT, it must have BAR=0. That is,

```
<bicond>
   <fs><f name=BAR><sym value=0></f></fs>
   <iff>
   <fs>
      <f name=N><any></f>
      <f name=V><any></f>
      <f name=SUBCAT><any></f>
   </fs>
</bicond>
```

The final constraint says that if a construction has a BAR value of 1 (i.e., it is a phrase), then the SUBCAT feature is irrelevant ( ). This is not biconditional, since there are other instances under which the SUBCAT feature is irrelevant. That is,

```
<cond><fs><f name=BAR><sym value=1></f></fs>
      <then>
```

```
      <fs><f name=SUBCAT><none></f></fs>
</cond>
```

The DTD fragment for feature structure constraints is as follows. Note that <cond> and <bicond> use the empty tags <then> and <iff>, respectively, to separate the antecedent and consequent. These are primarily for the sake of enhancing human readability.

```
<!-- 26.4:  Feature structure constraints                    -->
<!ELEMENT fsConstraints - -  (cond | bicond)*                  >
<!ATTLIST fsConstraints      %a.global;                        >
<!ELEMENT cond          - O  ((fs | f | fAlt), then, (fs | f |
                             fAlt))                           >
<!ATTLIST cond               %a.global;                        >
<!ELEMENT bicond        - O  ((fs | f | fAlt), iff, (fs | f |
                             fAlt))                           >
<!ATTLIST bicond             %a.global;                        >
<!ELEMENT iff           - O  EMPTY                             >
<!ATTLIST iff                %a.global;                        >
<!-- This fragment is used in sec. 26.1                      -->
```

26.5 A Complete Example

To summarize this chapter, the complete FSD for the example that has run through the chapter is reproduced below:

```
<!DOCTYPE teiFsd2 SYSTEM "teifsd2.dtd">
<teiFsd2>
<teiHeader>
<fileDesc>
<titleStmt>
   <title>A sample FSD based on an extract from Gazdar
          et al.'s GPSG feature system for English</title>
   <resp>
      <role>encoded by</role>
      <name>Gary F. Simons</name>
   </resp>
</titleStmt>
<publicationStmt>
This sample was first encoded by Gary F. Simons (Summer
Institute of Linguistics, Dallas, TX) on January 28, 1991.
Revised April 8, 1993 to match the specification of FSDs
in version P2 of the TEI Guidelines.
</publicationStmt>
<sourceDesc><p>
This sample FSD does not describe a complete feature
system.  It is based on extracts from the feature system
for English presented in the appendix (pages 245-247) of
Generalized Phrase Structure Grammar, by Gazdar, Klein,
Pullum, and Sag (Harvard University Press, 1985).
</sourceDesc>
</fileDesc>
</teiHeader>
<!-- ************************************************ -->
<fsDecl type=GPSG>
   <fsDescr>Encodes a feature structure for the GPSG analysis
     of English (after Gazdar, Klein, Pullum, and Sag)</fsDescr>
   <fDecl name=INV>
      <fDescr>inverted sentence</fDescr>
      <vRange>
          <vAlt><plus><minus></vAlt></vRange>
```

```
      <vDefault>
          <minus></vDefault>
   </fDecl>
   <fDecl name=CONJ>
      <fDescr>surface form of the conjunction</fDescr>
      <vRange>
          <vAlt>
              <sym value=and><sym value=both>
              <sym value=but><sym value=either>
              <sym value=neither><sym value=nor>
              <sym value=or><sym value=NIL>
          </vAlt></vRange>
      <vDefault>
          <none></vDefault>
      </fDecl>
   <fDecl name=COMP>
      <fDescr>surface form of the complementizer</fDescr>
      <vRange>
          <vAlt>
              <sym value=for><sym value=that><sym value=whether>
              <sym value=if><sym value=NIL>
          </vAlt></vRange>
      <vDefault>
          <if><fs><f name=VFORM><sym value=INF></f>
                  <f name=SUBJ><plus></f></fs>
           <then><sym value=for></if>
          </vDefault>
      </fDecl>
   <fDecl name=AGR>
      <fDescr>agreement for person and number</fDescr>
      <vRange><fs type=Agreement></fs></vRange>
      </fDecl>
   <fDecl name=PFORM>
      <fDescr>word form of a preposition</fDescr>
      <vRange><str rel=ne></str></vRange>
      </fDecl>
   <!-- The complete analysis includes additional features -->
   <fsConstraints>
      <cond><fs><f name=INV><plus></f></fs>
         <then>
            <fs>
               <f name=AUX><plus></f>
               <f name=VFORM><sym value=FIN></f>
            </fs></cond>
      <bicond><fs><f name=BAR><sym value=0></f></fs>
         <iff>
            <fs>
               <f name=N><any></f>
               <f name=V><any></f>
               <f name=SUBCAT><any></f>
            </fs></bicond>
      <cond><fs><f name=BAR><sym value=1></f></fs>
         <then>
            <fs><f name=SUBCAT><none></f></fs>
            </cond>
   </fsConstraints>
</fsDecl>
<!-- ************************************************ -->
<fsDecl type=Agreement>
   <fsDescr>This type of feature structure encodes the features
```

```
      for subject-verb agreement in English</fsDescr>
   <fDecl name=PERS>
      <fDescr>person (first, second, or third)</fDescr>
      <vRange><vAlt><sym value=1><sym value=2><sym value=3>
         </vAlt></vRange>
   </fDecl>
   <fDecl name=NUM>
      <fDescr>number (singular or plural)</fDescr>
      <vRange><vAlt><sym value=sg><sym value=pl>
         </vAlt></vRange>
   </fDecl>
</fsDecl>
</teiFsd2>
```

Chapter 27

Tag Set Documentation

This chapter describes an auxiliary DTD which may be used for the documentation of new elements, element classes and entities. It is primarily intended for use by those wishing to extend or modify the content of these Guidelines in a conformant manner, as described in chapters 29 ('Modifying the TEI DTD') on p. 619 and 28 ('Conformance') on p. 611; it may however be of use for the documentation of any SGML or SGML-like encoding scheme. The elements described here are those used to document the TEI scheme itself, in part VII of the current document.

Three distinct elements are used to document a tag set, the contents of each of which is described in more detail in the appropriate section of this chapter.

<tagDoc> documents the structure, content, and purpose of a single SGML element type. Attributes include:

usage specifies the optionality of an attribute or element. Legal values are:

req required

mwa mandatory when applicable

rec recommended

rwa recommended when applicable

opt optional

<classDoc> contains reference information for one element class; either elements which appear together in SGML content models, or elements which share some common attribute.

<entDoc> formally documents a single SGML entity within an encoding scheme.

In addition to these documentary elements, the following phrase-level elements may be found useful for marking up occurrences of element names etc. within the body of running text.

<gi> contains the name (generic identifier) of an SGML element. Attributes include:

TEI indicates whether this element is part of the TEI encoding scheme (i.e., defined in a TEI DTD fragment) or not. Legal values are:

yes this element is part of the TEI scheme.

no this element is not part of the TEI scheme.

<att> contains the name of an attribute appearing within running text. Attributes include:

TEI indicates whether this attribute is part of the TEI scheme (i.e., defined in a TEI DTD fragment) or not. Legal values are:

yes this attribute is part of the TEI scheme.

no this attribute is not part of the TEI scheme.

<val> contains a single attribute value.

<tag> contains text of a complete SGML start- or end-tag, possibly including attribute specifications, but excluding the opening and closing markup delimiter characters.

These four elements are included in the phrase-level elements available to any document using the auxiliary tag set defined in this chapter; to make them available to documents using other DTDs, an appropriate parameter entity should be defined.

As an example of the recommended use of these elements, we quote from an imaginary TEI working paper:

```
<p>The <gi>gi</gi> element is used to tag element names when
they appear in the text; the <gi>tag</gi> element however is
used to show how a tag as such might appear. So one might
talk of an occurrence of the <gi>blort</gi> element which had
been tagged <tag>blort type='runcible'</tag>. The <att>type</att>
attribute may take any name token as value; the default value is
<val>spqr</val>, in memory of its creator.
```

These elements and their components make up the auxiliary DTD for tag documentation, which is contained in the file *teitsd2.dtd*. This file has the following overall structure:

```
<!-- 27: File teitsd2.dtd: Auxiliary DTD for Tag Set          -->
<!-- Documentation                                            -->
<!-- Text Encoding Initiative: Guidelines for Electronic      -->
<!-- Text Encoding and Interchange. Document TEI P3, 1994.    -->

<!-- Copyright (c) 1994 ACH, ACL, ALLC. Permission to copy    -->
<!-- in any form is granted, provided this notice is          -->
<!-- included in all copies.                                  -->

<!-- These materials may not be altered; modifications to     -->
<!-- these DTDs should be performed as specified in the       -->
<!-- Guidelines in chapter "Modifying the TEI DTD."           -->

<!-- These materials subject to revision. Current versions    -->
<!-- are available from the Text Encoding Initiative.         -->

<!-- Embed entities for TEI generic identifiers.              -->

<!ENTITY % TEI.elementNames system 'teigis2.ent'               >
%TEI.elementNames;

<!-- Define entities for TEI keywords.                        -->

<!ENTITY % TEI.keywords.ent system 'teikey2.ent'               >
%TEI.keywords.ent;

<!-- Define element classes for content models, shared        -->
<!-- attributes for element classes, and global attributes.   -->
<!-- (This all happens within the file TEIclas2.ent.)         -->

<!ENTITY % TEI.elementClasses system 'teiclas2.ent'            >
%TEI.elementClasses;

<!-- Embed the core tag set                                   -->

<!ENTITY % TEI.core.dtd system 'teicore2.dtd'                  >
%TEI.core.dtd;

<!-- Define the top-level element for this DTD               -->

<!ELEMENT tsd           - O  ((tagDoc | entDoc | classDoc)+)   >
<!ATTLIST tsd                %a.global;                        >

<!-- Define some additions for the phrase level tags         -->

<!ELEMENT gi            - O  (#PCDATA)                        >
<!ATTLIST gi                 %a.global;
          TEI                (yes | no)          yes          >
<!ELEMENT tag           - -  (#PCDATA)                        >
<!ATTLIST tag                %a.global;
          TEI                (yes | no)          yes          >
```

```
<!ELEMENT att           - -  (#PCDATA)                          >
<!ATTLIST att                %a.global;
          TEI                (yes | no)          yes           >
<!ELEMENT val           - O  (#PCDATA)                          >
<!ATTLIST val                %a.global;                         >

<!-- Finally we define the elements specific to this DTD      -->

<!-- ... declarations from section 27.1                       -->
<!--     (The TagDoc element)                                 -->
<!--     go here ...                                          -->
<!-- ... declarations from section 27.1.1                     -->
<!--     (Attribute documentation)                            -->
<!--     go here ...                                          -->
<!-- ... declarations from section 27.2                       -->
<!--     (Element classes)                                    -->
<!--     go here ...                                          -->
<!-- ... declarations from section 27.3                       -->
<!--     (Entity Documentation)                               -->
<!--     go here ...                                          -->
```

27.1 The TagDoc Documentation Element

The <tagDoc> element is used to document an element type, together with its associated attributes. A completely specified <tagDoc> may comprise all of the following components in the order specified:

<rs> contains a general purpose name or referring string.
<gi> contains the name (generic identifier) of an SGML element.
<desc> contains a brief description of the purpose and application for an element, attribute, or attribute value.
<attList> contains documentation for all the attributes associated with this element, as a series of <attDef> elements.
<exemplum> contains a single example demonstrating the use of an element, together with optional paragraphs of commentary.
<eg> contains a single example demonstrating the use of an element or attribute. If the example contains SGML markup, it must be enclosed within a marked CDATA section.
<remarks> contains any commentary or discussion about the usage of an element, attribute, class, or entity not otherwise documented within the containing element.
<part> specifies the module or part to which a particular element, element class, or entity belongs in a modular encoding scheme such as the TEI.
<classes> specifies all the classes of which the documented element or class is a member or subclass.
<files> specifies the name of the operating system file(s) within which this markup component is declared.
<dataDesc> specifies the legal content of the element being documented, noting any semantic or application-dependent constraints, as well as constraints enforced by the SGML content model.
<parents> lists elements which can directly contain this element.
<children> lists the elements which this element may directly contain.
<elemDecl> contains the full form of an SGML declaration for the element documented using the reference concrete syntax.
<attlDecl> contains the SGML ATTLIST declaration associated with this element.
<ptr> defines a pointer to another location in the current document in terms of one or more identifiable elements.

<equiv> specifies an equivalent or comparable element in some other markup language.

As the content model for <tagDoc> makes clear, only the <gi>, <desc>, <exemplum>, <dataDesc>, <parents>, <children> and <elemDecl> elements are mandatory components. For elements bearing attributes, the <attList> and <attlDecl> components are also required for TEI conformance. For compatibility with the TEI system, use of the <classes> and <files> elements is strongly recommended. The only components of the <tagDoc> element which can appear more than once are the <exemplum>, <ptr> and <equiv> elements. The order of components may not be changed.

The <tagDoc> and its constituents are defined as follows:

```
<!-- 27.1:  The TagDoc element                                    -->
<!ELEMENT tagDoc        - -  (gi, rs?, desc, attList?,
                              exemplum*, remarks?, part?,
                              classes?, files?, dataDesc?,
                              parents?, children?, elemDecl,
                              attlDecl?, ptr*, equiv*)            >
<!ATTLIST tagDoc             %a.global;
          usage              (req | mwa | rec | rwa | opt)
                                                  opt             >
<!-- RS and PTR are defined in the core                           -->

<!-- GI is defined above                                          -->

<!ELEMENT desc          - O  (%paraContent)                       >
<!ATTLIST desc               %a.global;                           >
<!ELEMENT attList       - O  (attDef*)                            >
<!ATTLIST attList            %a.global;                           >
<!ELEMENT exemplum      - -  (p*, eg, p*)                         >
<!ATTLIST exemplum           %a.global;                           >
<!ELEMENT eg            - -  (#PCDATA)                            >
<!ATTLIST eg                 %a.global;                           >
<!ELEMENT remarks       - O  (%component.seq)                     >
<!ATTLIST remarks            %a.global;                           >
<!ELEMENT part          - O  (#PCDATA)                            >
<!ATTLIST part               %a.global;
          type               CDATA               #IMPLIED
          name               CDATA               #IMPLIED       >
<!ELEMENT classes       - O  (#PCDATA)                            >
<!ATTLIST classes            %a.global;
          names              CDATA               #REQUIRED      >
<!ELEMENT files         - O  EMPTY                                >
<!ATTLIST files              %a.global;
          names              CDATA               #IMPLIED       >
<!ELEMENT dataDesc      - O  (%phrase.seq)                        >
<!ATTLIST dataDesc           %a.global;                           >
<!ELEMENT parents       - O  (#PCDATA)                            >
<!ATTLIST parents            %a.global;                           >
<!ELEMENT children      - O  (#PCDATA)                            >
<!ATTLIST children           %a.global;                           >
<!ELEMENT elemDecl      - O  (#PCDATA)                            >
<!ATTLIST elemDecl           %a.global;                           >
<!ELEMENT attlDecl      - -  (#PCDATA)                            >
<!ATTLIST attlDecl           %a.global;                           >
<!ELEMENT equiv         - O  (%specialPara)                       >
<!ATTLIST equiv              %a.global;
          scheme             CDATA               #REQUIRED      >
<!-- This fragment is used in sec. 27                             -->
```

27.1.1 The AttList Documentation Element

The <attList> element is used to document information about a collection of attributes, either within a <tagDoc>, or within a <classDoc>. It consists of a series of <attDef> elements, each documenting a single attribute and each using an appropriate selection from the following elements:

<attDef> contains the definition of a single attribute. Attributes include:

usage specifies the optionality of an attribute or element. Legal values are:

req required

mwa mandatory when applicable

rec recommended

rwa recommended when applicable

opt optional

<attName> contains the name of the attribute being defined by an <attDef> element.

<rs> contains a general purpose name or referring string.

<desc> contains a brief description of the purpose and application for an element, attribute, or attribute value.

<dataType> specifies an SGML *declared value* for an attribute.

<valList> contains a list of value and description pairs for an attribute. Attributes include:

type specifies the extensibility of the list of attribute values specified. Legal values are:

closed only the values specified are permitted.

semi all the values specified should be supported, but other values are legal and software should have appropriate fallback processing for them.

open the values specified are sample values only.

<val> contains a single attribute value.

<valDesc> specifies any semantic or syntactic constraint on the value that an attribute may take, additional to the information carried by the <dataType> element.

<default> specifies the default declared value for an attribute.

<eg> contains a single example demonstrating the use of an element or attribute. If the example contains SGML markup, it must be enclosed within a marked CDATA section.

<remarks> contains any commentary or discussion about the usage of an element, attribute, class, or entity not otherwise documented within the containing element.

<equiv> specifies an equivalent or comparable element in some other markup language.

It will be noted that several of these elements are used identically to document both elements and attributes. Specific to attributes are <dataType>, <valList>, <valDesc>, <val> and <default>. For any attribute documented in this way, either a <valList> or a <valDesc> must be supplied to specify the range of permitted values for an attribute. A <valList> should be used if the intended set of values can be enumerated; a <valDesc> if it cannot. A legal <attDef> specification must contain an <attName>, a <desc>, a <datatype>, either a <valList> or a <valDesc>, and a default; the other elements listed above are all optional.

The <attList> within a <tagDoc> is used to specify only the attributes which are specific to that particular element. Attributes which are shared by other elements, or by all elements, should be documented by an <attList> contained within a <classDoc> element, as described in section 27.2 ('Element Classes') on p. 606 below.

The following <attList> demonstrates some of the possibilities; for more detailed examples, consult the tagged version of the reference material in these Guidelines.

```
<attDef usage=opt>
<attName>type
<desc>describes the form of the list.
<datatype>CDATA
<valList type=semi>
    <val>ordered<desc>list items are numbered or lettered.
    <val>bulleted<desc>list items are marked with a
```

```
      <soCalled>bullet</soCalled> or other typographic device.
    <val>simple<desc>list items are not numbered or bulleted.
    <val>gloss<desc>each list item glosses some term or
      concept, which is given by a <gi>label</gi> element preceding
      the list item.
</valList>
<default>simple
<remarks><p>The formal syntax of the element declarations allows
<gi>label</gi> tags to be omitted from lists tagged <tag>list
type=gloss</tag>; this is however a semantic error.
</attDef>
```

Those elements from the above list which are unique to attributes are declared as follows:

```
<!-- 27.1.1:  Attribute documentation                          -->
<!ELEMENT attDef        - O  (attName, rs?, desc, (dataType,
                              (valList | valDesc)?), default,
                              eg?, remarks?, equiv*)            >
<!ATTLIST attDef             %a.global;
          usage              (req | mwa | rec | rwa | opt)
                                                  opt           >
<!ELEMENT attName       - O  (#PCDATA)                           >
<!ATTLIST attName            %a.global;                          >
<!ELEMENT dataType      - O  (#PCDATA)                           >
<!ATTLIST dataType           %a.global;                          >
<!ELEMENT valList       - -  ((val, desc)*)                      >
<!ATTLIST valList            %a.global;
          type               (closed | semi | open)
                                                  open          >
<!ELEMENT valDesc       - O  (%phrase.seq)                       >
<!ATTLIST valDesc            %a.global;                          >
<!ELEMENT default       - O  (#PCDATA)                           >
<!ATTLIST default            %a.global;                          >
<!-- This fragment is used in sec. 27                          -->
```

27.2 Element Classes

The element **<classDoc>** is used to document an *element class*, as defined in section 3.7 ('Element Classes') on p. 51. It has the following components:

<classDoc> contains reference information for one element class; either elements which appear together in SGML content models, or elements which share some common attribute. Attributes include:

type indicates whether this is an model class, an attribute class, or both. Sample values include:

model members of this class appear in the same SGML content models

atts members of this class share common attributes

both members of this class share attributes and appear in the same SGML content models

<class> specifies the name of an element class.

<rs> contains a general purpose name or referring string.

<desc> contains a brief description of the purpose and application for an element, attribute, or attribute value.

<attList> contains documentation for all the attributes associated with this element, as a series of <attDef> elements.

<remarks> contains any commentary or discussion about the usage of an element, attribute, class, or entity not otherwise documented within the containing element.

<part> specifies the module or part to which a particular element, element class, or entity belongs in a modular encoding scheme such as the TEI.

<classes> specifies all the classes of which the documented element or class is a member or subclass.

<ptr> defines a pointer to another location in the current document in terms of one or more identifiable elements.

<equiv> specifies an equivalent or comparable element in some other markup language.

Of these elements, only the <class> and <desc> elements are required components. If present, the other elements must be given in the order specified, and only <ptr> and <equiv> may be repeated.

The attribute type is used to distinguish between "model" and "attribute" classes; for the former, a <classDoc> simply exists so that members of the class it documents may point to it (by specifying the value of its id attribute among the values specified by the names attribute of their <classes> component); for the latter, the <classDoc> additionally contains an <attList> which specifies the attributes shared by the members of the class. A class may perform both functions, of course.

Where a class inherits properties or attributes from some other class, the <classes> element may be used to indicate this fact. Membership of an attribute class can be inherited by any class, but model-only classes may not include attribute-only classes amongst their members. For further discussion of the TEI class system, see section 3.7 ('Element Classes') on p. 51.

The <classDoc> element and the elements unique to it are declared as follows:

```
<!-- 27.2:  Element classes                                    -->
<!ELEMENT classDoc      - O  (class, rs?, desc, attList?,
                              remarks?, part?, classes?, files?,
                              ptr*, equiv*)                      >
<!ATTLIST classDoc           %a.global;
          type               (model | atts | both)
                                                #IMPLIED       >
<!ELEMENT class         - O  (#PCDATA)                          >
<!ATTLIST class              %a.global;                         >
<!-- all other constituents are defined above                  -->

<!-- This fragment is used in sec. 27                          -->
```

27.3 Entity Documentation

The <entDoc> element is used to document any other entity not otherwise documented by the elements described in this chapter. Its chief uses are to provide systematic documentation for parameter entities used within TEI DTD fragments (for example, those used to enable different components of the TEI DTD, or to describe common content models), but it may be used for any purpose. It has the following components:

<entDoc> formally documents a single SGML entity within an encoding scheme. Attributes include:

type indicates whether this is a general or a parameter entity. Legal values are:

pe parameter entity

ge general entity

<entName> contains the full name of an entity, excluding the percent sign in the case of a parameter entity.

<rs> contains a general purpose name or referring string.

<desc> contains a brief description of the purpose and application for an element, attribute, or attribute value.

<remarks> contains any commentary or discussion about the usage of an element, attribute, class, or entity not otherwise documented within the containing element.

<string> contains the intended expansion for the entity documented by an <entdoc> element, enclosed by quotation marks.

<ptr> defines a pointer to another location in the current document in terms of one or more identifiable elements.

<equiv> specifies an equivalent or comparable element in some other markup language.

Of these, only <entName>, <desc> and <string> are required components. If present, the other elements must be given in the order specified, and only <ptr> and <equiv> may be repeated.

The <entDoc> element and the elements unique to it are declared as follows:

```
<!-- 27.3:  Entity Documentation                                  -->
<!ELEMENT entDoc        - -  (entName, rs?, desc, remarks?,
                              string, ptr*, equiv*)               >
<!ATTLIST entDoc             %a.global;
          type               (pe | ge)           #REQUIRED       >
<!ELEMENT entName       - O  (#PCDATA)                            >
<!ATTLIST entName            %a.global;                           >
<!ELEMENT string        - -  (#PCDATA)                            >
<!ATTLIST string             %a.global;                           >
<!-- All other constituents are defined above                    -->

<!-- This fragment is used in sec. 27                            -->
```

Part VI

Technical Topics

Chapter 28

Conformance

The notion of *TEI conformance* is intended as an aid in describing the format and contents of a particular document or set of documents. These concepts are expected to be useful in:

- agreements for the interchange of documents among researchers
- agreements for the deposit of texts in archives and their distribution from archives
- describing the documents to be produced by or for a given project
- defining the classes of documents accepted or rejected by a given piece of software

This chapter describes the areas in which these terms are defined and specifies their meaning. It also proposes other terms for related concepts and points out some dangers in the careless use or application of these terms.

28.1 Definitions of Terms

The terms described here should be considered technical terms for users and implementors of the TEI Guidelines and should be used only in the senses given and with the usages described.

28.1.1 TEI-Conformant Document

A document is *TEI conformant* if it is either in *TEI local processing format* or in *TEI interchange format*. A full description of the document should specify which format it is in. The term *TEI conformance* does not apply to software: programs can be usefully described as *accepting* or *validating* TEI-conformant documents or some subset of TEI-conformant documents, but the TEI defines no required processing model against which software could be measured. Programs are thus not themselves conformant or non-conformant and should not be so described.

28.1.2 TEI Local Processing Format

A document is in *TEI local processing format* if

- it is a conforming SGML document with a legal SGML declaration;
- it uses the document type declarations provided by the TEI, either without modifications or with all modifications effected by inclusion in the DTD subset as described in section 28.3 ('Modifications to TEI Document Type Declarations') on p. 613;
- all modifications to meaning or use of defined tags, and all new tags, are documented in TEI Tag Set Declarations which accompany the document, as defined in chapter 27 ('Tag Set Documentation') on p. 601;
- it includes, in the TEI header, all the elements required by the TEI declarations for the TEI header;
- it contains no markup other than SGML or declared notations for graphics, tables, figures, etc. That is, unless a declared notation is in use, the semantics of any content character in the document are exhausted by its identity as a graphic character.

A TEI-local-processing-format document may be described as requiring *DTD extensions* if it modifies the TEI-supplied DTDs or SGML prolog in any of the ways described under 28.3 ('Modifications to TEI Document Type Declarations') on p. 613. The following terms are synonymous: *document in TEI local processing format*, *TEI local-processing document*, and *TEI local-processing-conformant document*.

28.1.3 TEI Interchange Format

A document is in *TEI interchange format* if it conforms to the TEI local-processing format and if further:

- its SGML declaration is either the predefined SGML declaration for TEI interchange documents or an SGML declaration which differs from it only in ways allowed by section 28.2 ('Modifications to TEI SGML Declaration') on p. 613;
- it makes no use of any of the following SGML constructs:
  - short references
  - the RANK feature
  - omission of generic identifiers in start- and end-tags [1]
  - keywords other than INCLUDE, IGNORE, and CDATA on a marked section
- it includes no SUBDOC subordinate document by means of an entity reference embedded directly within content data (SUBDOCs must be included by giving the entity reference as the value of an attribute);
- it does not provide different definitions for the same entity in different document types.

A TEI-interchange-format document may be described as requiring *DTD extensions* if its DTD is modified in any of the ways described in section 28.3 ('Modifications to TEI Document Type Declarations') on p. 613. The following terms are synonymous: *document in TEI interchange format*, *TEI interchange document*, and *TEI interchange-conformant document*.

28.1.4 TEI Packed Interchange format

A document is in TEI packed interchange format with a given *transmission character set* and a given *transmission entity set* if all of the following are true:

- all separate entities in the document are packed into a single entity (file) in a manner conforming to ISO 9069 (SDIF) or to some other TEI-authorized form;
- all characters occurring in SGML names (generic identifiers and attribute names) occur within the transmission character set;
- all characters in the document content and attribute values either occur within the transmission character set or are represented by an appropriate entity reference using an entity name included in the transmission entity set;
- the transmission character and entity sets are named in the header of the packed file and in any accompanying paper documentation.

< Any pre-transmission processing required to convert a document to meet the above requirements for conformance to the TEI-interchange-format is called *packing*. With prior agreement between parties to an exchange, interchange documents may use character code set switching as defined in ISO 2022, its national analogues, or successor standards. A full description of a document in TEI packed interchange format *must* specify the transmission character set and the transmission entity set used in the document.

28.1.5 TEI Recommended Practice

A document follows *TEI recommended practice* if:

- it is a TEI-conformant document;
- wherever the guidelines say to prefer one encoding practice to another, the preferred practice is used;

[1] This is one of several abbreviations allowed by the SHORTTAG feature; the others (omission of attribute names under certain circumstances and omission of non-required attribute values) are allowed.

- all textual features which the guidelines recommend be captured are in fact encoded.

28.1.6 TEI Abstract Model

A document follows the *TEI abstract model* if it tags the features specified in the TEI documentation and documentation, and their structural interrelations agree with those specified in the TEI DTDs.

28.2 Modifications to TEI SGML Declaration

The SGML declaration for TEI interchange documents may differ from that provided in TEI documentation in these ways:

- the CHARSET clause must be used to define the transmission character set (possibly in connection with the SHUNCHAR specification in the SYNTAX clause);
- the CAPACITY clause may be used to raise (but not lower) capacities;
- the SYNTAX clause may be used to define the SGML syntax used in the document. Notably:
  - the SHUNCHAR specification within the SYNTAX clause may be used to restrict the transmission character set;
  - the BASESET and DESCSET specifications within the SYNTAX clause must be used to describe the transmission character set;
  - the DELIM and NAMES specifications may be used to modify the SGML syntax.
- in the FEATURES clause, CONCUR may be set to NO if concurrent markup is not used in the document.

The following portions of the SGML declaration may not be modified in TEI interchange documents:

- the CAPACITY and QUANTITY values may be increased but not decreased;
- the SCOPE clause may not be changed;
- no new FEATURES may be turned on.

The SGML declaration for TEI-local-format documents may be modified without restriction. Some recommendations for usage are made in document TEI P1, but these recommendations are not normative.

28.3 Modifications to TEI Document Type Declarations

A TEI-conformant document (whether for local processing or for interchange) may make any change to the TEI-supplied document type declarations which is allowed by SGML and the controlling SGML declaration. All such changes should be effected within the SGML DTD subset. For further discussion, see chapter 29 ('Modifying the TEI DTD') on p. 619. In all TEI-conformant documents, the document type name must be <tei>, followed by a digit indicating the currently released version number for the TEI guidelines in use. The associated system data should refer to a file containing the unmodified TEI main DTD fragment, as in the following example:

```
<!DOCTYPE tei2 system "tei2.dtd" [
<!-- local modifications here -->
]>
```

The following must remain true of the DTD after modification:

- the overall document must contain a single <teiHeader> element and a single <text> element, in that order; in the case of a corpus or collection the overall collection may have a <teiHeader> followed by a series of <tei1> documents;

- the <teiHeader> element must include elements for:

  title statement the title of the machine-readable work and the names of those responsible for it;

  publication statement place and date of publication or distribution of the machine-readable document;

  source description bibliographic description of the copy text or source of the electronic text, including at least title, author, and edition.

A TEI-conformant document may be said to require *DTD extensions* if it:

- defines new elements;
- modifies the content model, declared content, or omissibility of any element;
- adds or modifies any attribute definitions;
- renames any elements, attributes, or attribute values;
- defines any new document types;
- declares any non-SGML notations.

Without requiring DTD extensions, therefore, any TEI document may:

- define entities and parameter entities;
- include processing instructions and comments in its DTD subset.

For local-processing purposes, a TEI document may also, without requiring DTD extension:

- include link type declarations etc. in its SGML prolog;
- define and use short reference mapping in its DTD.

Note that TEI interchange documents may *not* include link type or short reference declarations because the SGML declaration for interchange does not allow them. It is expected that the notion of DTD extension will be particularly useful in describing the classes of documents accepted or validated by software.

28.4 TEI Processing Model

This section is included for illustrative purposes only; it does not restrict the processing of TEI or other documents. It simply distinguishes a number of typical ways in which a project may choose to apply the TEI Guidelines to different kinds of processing.

28.4.1 Document Capture and Reclamation

First, data might be captured by keyboarding into a locally defined data capture format, or by scanning into a locally defined scanner-file format. From these initial forms, transducers might convert the files into a standard local storage format.

28.4.2 Local Storage Format and Application Software

The local storage format might be the input format of some application program used frequently by the project. In this case, transducers might be necessary to prepare data for processing by other applications. Alternatively, the local storage format might be independent of the formats used by application programs; transducers would be needed to prepare data for any processing. Such an independent format is useful if the local storage format needs to contain more information than any single application can conveniently handle. The local storage format might be SGML-conformant without being TEI-conformant, e.g. because it uses local DTDs instead of the standard TEI DTDs, or because it uses a TEI local processing format. Local software may be used to validate a TEI local-processing format, to transduce documents into the input formats needed by applications, and when appropriate to transform documents into the TEI interchange format for exchange with other sites.

Finally, the TEI interchange format may be used as a local storage format. It is not expected that this will be a very common practice, since it is expected that most sites interested in TEI conformance will eventually acquire SGML-conformant software which allows for a more

compact local storage than does the interchange format. In the absence of SGML software, however, some projects may find the TEI interchange format (or perhaps a restrictive variant of it) useful, because such a format can be relatively easy to parse with ad hoc software.

Whether the local storage format is strictly TEI conformant or not, it may follow TEI-recommended practice in its selection of textual features to be marked up, in its tag names, in its documentation practices, etc.

28.4.3 Enrichment and Other Processing

Over the course of the project, analysis and processing may result in interim results which may be incorporated into the locally stored copy of the text so that the interim results can be used in later processing. This process of enrichment can be carried out either by manual editing of the documents using conventional text editors, or by application programs.

28.4.4 Data Export

When a document is to be exchanged with another site using the TEI interchange format, it must first be transduced from the local storage format to TEI interchange form. If local documents are already TEI-conformant, this requires either no processing at all, or a relatively simple normalization which can be handled readily by the normalization facilities of most SGML parsers. If the local storage form is non-SGML conformant, some transducer must be used to transform it into the TEI interchange format. The TEI-interchange-format document must then be packed for shipping into the TEI packed interchange format, using a packing program. This program will gather the constituent parts (files) of a document into a single file, and ensure that the file contains no characters whose safe passage to the recipient of the data is endangered by the transmission path. If the ultimate recipient of the document is unknown, the set of safe characters is very small. The specific *transmission character set* however is independent of TEI conformance: any convenient set may be used where both parties agree. The packer will ensure that the transmission character set is properly identified.

28.4.5 Data Import

When a document is received from another site using the TEI packed interchange format, it must first be unpacked into a TEI interchange-format document in the local character set. It may then be necessary to *naturalize* it by translating it into the local storage format; if the local format is TEI- or SGML-conformant, no processing is needed (although some SGML processors may offer a facility for suppressing omissible markup).

28.4.6 TEI Conformance in the Processing Model

The notions of TEI interchange format and TEI packed interchange format are central to the exchange of documents using the TEI guidelines, whether the local storage format is TEI-conformant or not. The TEI interchange format and the TEI local-processing format may each be used as a local storage format, though the local storage format might well differ from either of these without materially affecting the use of TEI formats for interchange. The TEI interchange format being less flexible than the local-processing format, it is expected that sites using SGML-conformant software may use the latter, while sites without such software may prefer the former. The notion of TEI recommended practice, it is hoped, will be relevant to decisions about what textual features should be recorded during data capture and will thus affect data-capture formats and the transducers which render captured files into the local storage format. The TEI abstract structure may be useful in developing local non-SGML markup schemes for data capture or for processing with ad hoc application programs. It is strongly recommended that the TEI recommendations, as well as the TEI abstract structure, be used for such development as well.

28.5 Aspects of Conformance and Document Description

28.5.1 Character Sets

Neither the character sets used for local processing nor those used for transmission of interchange documents are restricted by the definition of TEI conformance. For local processing, users will typically use the system character set of their local system or some modification thereof. For exchange with known partners, users should choose any convenient character set; typically the most convenient is the set of all characters which:

- are transmitted successfully over the existing transmission link;
- occur in both sender's and receiver's local coded character sets.

For blind exchange with unknown partners a conservative choice of transmission set is needed to ensure that characters arrive correctly. How conservative the choice need be depends on the medium of transmission. The *ISO 646 subset* defined in section 4.3 ('Character Set Problems in Interchange') on p. 75 remains the only guaranteed safe set of characters for the regional and international networks most widely used, although larger character sets are increasingly coming into use. This is largely because silent and not always reversible translation between character sets remains a feature of transmissions across current disparate networks. At the present time (1993) therefore, only the ISO 646 subset is recommended for fully blind interchange, although the full complement of ISO character sets may be used successfully in some subdomains. In transmission by disk or tape, however, no silent translation is likely to occur, and so larger sets may be successfully used in blind interchange. The primary danger is a failure of software in the receiving machine to process the characters correctly; at this time (1991), ASCII or 94-character U.S. EBCDIC appear to represent the largest safe choices; other national character sets may of course be used if good internal documentation is also provided. Note that the transmission character set does not associate specific binary encodings with the characters in the set. In the technical senses, it is a character set, not a *coded* character set. This means that a document may undergo various automatic translations from one coded character set to another (notably, in the case of transmission over international networks, from ASCII to EBCDIC or vice versa) without leaving the *transmission character set*. For further discussion of the topics addressed in this section, reference should be made to chapter 30 ('Rules for Interchange ') on p. 627.

28.5.2 SGML Declaration

The utility of various SGML constructs is discussed in section 2.2 of document TEI P1 version 1. The restrictions on SGML declarations and SGML usage in TEI interchange documents discussed above under 28.2 ('Modifications to TEI SGML Declaration') on p. 613 are derived from that discussion. No restrictions are made on SGML usage in the local processing format because such usage is best determined locally and has no impact on interchange.

28.5.3 SGML Document Type Declaration

The document type declaration provided by the TEI is intended to cover as wide a variety of document types and processing needs as proved feasible. It is impossible, however, for any finite list of text elements to cover every need of textual research and processing. As a result, extension of the TEI DTD has no effect on strict TEI conformance, as long as certain restrictions are observed; these have the effect of ensuring that later users of a file can easily see what changes have been made to the DTDs and what the new tags are intended to mean. The requirement that all new or modified tags be documented, however, is formally verifiable only to a limited extent. It is possible for a program to verify that for every tag introduced in a DTD modification, a corresponding record exists in a Tag Set Declaration. It is impossible, however, to verify using formal means that the entry in the tag set declaration makes sense. Purely formal conformance measures, therefore, must be supplemented with human inspection of the documentation. The concept of DTD extension is introduced to allow the concise description of software which is

designed to handle documents encoded using the published DTDs but which is not prepared to deal with tags not included there. [2] All sections of the TEI DTD are subject to modification by the user, except that a documentary header must be provided and distinguished from the text itself, and that documentary header must include tagged elements identifying the document encoded and those responsible for the encoding. This ensures that all TEI-conformant documents will have at least this bare minimum of accompanying documentation.

28.5.4 Tag Usage and Feature Marking

The basic design principles of the TEI require the notion of *TEI conformance* to be applicable to existing electronic documents if they are translated into a proper format, without requiring the insertion of information not captured in the initial preparation of the text. [3] At the same time, the TEI is charged with formulating advice to those engaged in the creation of new electronic texts and is required to distinguish what is actively recommended for general use from what is merely optional, provided for use by those engaged in a particular sort of work. The notion of *TEI recommended practice* is introduced to allow the concise description of documents in which not only the requirements, but also the recommendations of the Guidelines are followed. It is hoped that while projects to convert existing electronic data may content themselves with achieving TEI conformance, projects to produce new electronic texts will produce documents following TEI recommended practice. To distinguish those projects which follow the TEI's recommendation to use SGML markup from those which capture the same underlying textual features but do so using non-SGML markup, the notion of the *TEI abstract model* is introduced; it is this which a non-SGML-based encoding can have in common with the TEI.

28.5.5 Non-SGML Markup

In exchanging texts for use by others, the goal of an interchange format is to ensure that the information encoded in an electronic version of a text can be correctly understood and processed by the recipient as well as by the originator of the text. To assure the achievement of this goal, the definition offered here of TEI conformance restricts markup in TEI conformant documents to SGML markup and to properly declared non-SGML notations. The latter are explicitly recommended for the encoding of tables, figures, etc. and so cannot reasonably be excluded. Since they do place a burden on the recipient for proper processing, the use of any such non-SGML notation is defined to fall within the class of DTD extensions. Because of the escape clause for graphics, etc., it is in principle possible to create a TEI conformant document by embedding a document using any arbitrary markup into a driver file containing a TEI header and a declaration for the appropriate markup as a non-SGML notation. Though it falls within the letter, such a practice falls outside the spirit of TEI-conformant document interchange.

[2] Some will regard such simplifications as useful ways of making it easier to develop software which accepts TEI-conformant documents; others will deplore the failure of such software to accept all TEI-conformant documents including those which extend the TEI DTD. In providing the notion of *DTD extension* for describing what documents are and are not accepted by such software, the TEI acts in the belief that such software will in fact be developed; it neither endorses nor deplores its construction or use.

[3] See document TEI PC P1 "The Preparation of Text Encoding Guidelines."

Chapter 29

Modifying the TEI DTD

These Guidelines provide an encoding scheme suitable for encoding a wide range of texts, and capable of being extended in well-defined and documented ways for texts that cannot be conveniently or appropriately encoded using what is provided. This chapter describes how the TEI encoding scheme may be modified and extended.

The formal mechanisms provided by these Guidelines are syntactic mechanisms for encoding information in electronic texts. The semantics associated with these syntactic mechanisms are described only informally in the accompanying text. Although the descriptions have been written with care, there will inevitably be cases where the intention of the contributors has not been conveyed with sufficient clarity to prevent users of the Guidelines from "extending" them in the sense of attaching slightly variant semantics to them.

Beyond this unintentional semantic extension, some of the elements provided can intentionally be used in a variety of ways; for example, the element <note> has an attribute type that can take on arbitrary string values, depending on how it is used in a document. A new type of annotation, therefore, requires no extension to the TEI document type definition.

Furthermore, there are several ways for combining and extending the existing syntactic mechanisms themselves. Earlier chapters have identified these:

- Combining the supplied tag sets—the core with one or more base tag sets and additional tag sets—as described in chapter 3 ('Structure of the TEI Document Type Definition') on p. 35
- Documenting how languages are represented using character sets in a document by providing one or more writing system declarations, as described in chapter 25 ('Writing System Declaration') on p. 571
- Extending the intentionally open-ended feature structure mechanism by providing one or more feature system declarations, as described in chapter 26 ('Feature System Declaration') on p. 589

This chapter explains how the supplied tag sets can be modified by suppressing elements, renaming elements, extending syntactic classes, and adding elements. The different techniques described here have different effects on the level of TEI conformance to be ascribed to a text, as described in chapter 28 ('Conformance') on p. 611.

The TEI DTD is designed to support modification of the tag sets in a documented way that can be validated by an SGML parser. Those wishing to modify the tag sets must do so using *only* the mechanisms described in this chapter if the resulting documents are to be TEI conformant.

This approach to modification implies the existence of two versions of any DTD derived from the TEI tag sets. The first version is a fixed human-readable version. The TEI DTD fragments that have appeared earlier in these Guidelines are all presented in this version, save in the few cases where forward pointers to this chapter appear. This is the *publication version* of the DTD. Applications that do not use any of the modification mechanisms described in this chapter should use these DTD fragments.

The second version of a TEI DTD is automatically derived from the first by the introduction of parameter entities. These parameter entities are given values that contain small parts of the definition of the DTD. If these parameter entities are not modified, their default expansion

(equivalent to the publication version) is used within the DTD. However, their definitions can be easily changed for a specific application. This is the *modifiable version* of the DTD. It is also directly usable by an SGML parser.

In the absence of any modification, the TEI core DTD and the additions to it embodied in the base and additional tag sets behave as follows:

1. All the elements described in the relevant sections of these Guidelines are defined.
2. The names of these elements are as given by these Guidelines.
3. The content model of each element is as given by these Guidelines.
4. The attributes of each element and the names and types of these attributes are as given in these Guidelines.
5. The membership of element classes, and the resultant inheritance of attributes are as given in these Guidelines.

The modification mechanisms allow the user to override these defaults in the following ways, while retaining conformance to the TEI Guidelines:

1. The definition of elements may be suppressed, so deleting the associated elements from the modified DTD.
2. The name (generic identifier) associated with an element may be changed, while preserving the semantics of the element. Note, however, that the new name may not clash with the default name of any element defined in these Guidelines.
3. The global teiform attribute should be used to specify the TEI name for such renamed elements.
4. Those parts of the content model of an element which are specified by classes may be extended by adding members to the classes.
5. Further attributes, together with their names and types, may be specified for an individual element and existing attributes for an individual element may be renamed. Note that the new names may not clash with names of existing attributes for the element.
6. Further attributes, together with their names and types, may be specified for the elements in a particular class and those inheriting characteristics from that class.

The modification mechanisms presented in this chapter are quite general, and may be used to make all the types of changes just listed. They can also be used to make more complete modifications of the DTD; if changes other than those listed above are made, however, the resulting text will no longer be TEI conformant.

These Guidelines provide for user modification of the TEI DTD largely by using SGML parameter entities at appropriate points in the DTD. It is not absolutely essential to understand them in detail to modify the TEI DTD (the examples later in this chapter can be followed cookbook fashion), but it will probably prove helpful.

Parameter entities are an SGML mechanism for allowing string substitution within markup declarations; they can thus can be used to effect changes in declarations.

A parameter entity, unlike an element, may be declared more than once in an SGML document type definition; if more than one declaration is given, the parser uses the first one it encounters. Since the declaration subset within the document is read before the external file containing the predefined DTD, an entity declaration in the DTD subset will take precedence over one in the external file. In the TEI DTD, the literal string which defines the model group for some elements, say <p>, is made the value of a parameter entity; the actual element declaration for <p> contains not the literal string itself, but a reference to the parameter entity (in this case, *paraContent*). If the document's DTD subset contains a declaration for *paraContent*, this will be used in preference to the standard definition within the external TEI DTD files. The redefinability of parameter entities accounts both for the TEI's use of parameter entities as the vehicle for effecting extensions, and for the separation of entity definitions from other material (to be defined below) that might be needed for certain modifications of the TEI DTD.

Local modifications are most conveniently grouped into two files, one containing modifications to the TEI parameter entities, and the other containing new or changed declarations of elements and their attributes. Names for these files should be specified by the parameter entities *TEI.extensions.ent* and *TEI.extensions.dtd*, by declarations such as the following:

```
<!ENTITY % TEI.extensions.ent system 'project.ent' >
<!ENTITY % TEI.extensions.dtd system 'project.dtd' >
```

These declarations must be given in the document's DTD subset. The first will be needed for all of the modifications discussed below. The second is only required for the last type of extension described. The parameter entities thus defined are then referenced at an appropriate point in the compiling of the TEI DTD, as further described in section 3.3 ('Invocation of the TEI DTD') on p. 39.

29.1 Kinds of Modification

There are several kinds of modification that can be made to the TEI DTD as follows:

1. deletion of elements,
2. renaming of elements,
3. extension of classes, and
4. modification of content models or attribute lists.

These are described in the remaining sections of this chapter.

Each kind of modification changes the set of documents that can be parsed using the DTD. Each combination of the original TEI DTD fragments may be thought of as defining a certain set of documents. Each DTD resulting from a modified set of TEI DTD fragments allows a different set of documents to be parsed. The set of documents parsed by the original DTD may be properly contained in the set of documents parsed by a modified DTD, or vice versa. Modifications that have either of these two results are called *clean modifications* in the remainder of the chapter. Alternatively, the set of documents parsed by the original DTD might overlap the set of documents parsed by the modified DTD with neither being properly contained in the other. Modifications that have this result are called *unclean modifications* in the remainder of the chapter.

29.1.1 Suppressing Elements

The simplest way to modify the supplied tag sets is to suppress one or more of the supplied elements. In the modifiable version of the DTDs, every element declaration is enclosed by a marked section. The marked section is governed by one of the keywords `IGNORE` or `INCLUDE`, which is provided indirectly using a parameter entity. This parameter entity has the same name as the generic identifier of the element. Thus, the declaration for the paragraph element, `<p>`, occurs within a marked section "guarded" by the parameter entity *p*:

```
<![ %p [
<!-- element and attlist declaration for p goes here   -->
]]>
```

The declaration given for these *guarding entities* in the modifiable version is `INCLUDE` in all cases. The construct above is interpreted thus: the first of the three lines is the opening of a marked section; when the parser encounters the section and sees the keyword `INCLUDE` as its guard (more precisely, sees a parameter entity the value of which is the keyword `INCLUDE`), the content of the marked section is parsed; the second line of the three is the content of the marked section; and the third line of the three is the closing of a marked section. If the guard is changed to `IGNORE`, the SGML parser will ignore the content of the marked section.

Thus, to delete the declaration of a generic identifier and thus suppress the element entirely, the entity that provides the guard on the marked section wherein the element declaration appears must simply be set to `IGNORE`. For example, if the `<note>` element is not to be used in a particular application, the line

```
<!ENTITY % note 'IGNORE'>
```

must be inserted in the file which has its name given by the entity *tei.extensions.ent*.

Two different cases of deleting one or more elements from the TEI DTD can be identified. The first case involves deleting only elements that are optional wherever they appear in TEI

documents. Deleting these is clean in the sense that documents that are parseable with the modified DTD can also be parsed according to the original TEI DTD. To say this another way, the set of documents matching the new DTD is contained in the set of documents matching the original DTD.

The second case involves deleting elements that are required in one or more of their appearances in TEI documents. Deleting these is unclean in that some documents that can be parsed according to the new DTD could not be parsed according to the original TEI DTD. To say this another way, the set of documents matching the new DTD neither contains nor is contained in the set of documents matching the original DTD.

29.1.2 Renaming Elements

In the modifiable version of the TEI DTD, elements are not referred to directly by their generic identifiers; instead, the modifiable version of the DTD makes use of parameter entities that expand to the standard generic identifiers. This allows renaming of elements by redefining the appropriate parameter entities. The names of parameter entities used for naming are formed by taking the standard generic identifier of the element and attaching the string "n." (for "name") as a prefix. Thus, the standard generic identifiers for paragraphs, notes and quotations, <p>, <note>, and <q>, are defined by declarations of the following form:

```
<!ENTITY % n.p    'p'>
<!ENTITY % n.note 'note'>
<!ENTITY % n.q    'q'>
```

These parameter entities are all contained within a file (*teigis.ent*) which is embedded during the compilation of a TEI DTD. To change the name of an element therefore, all that is needed is to provide an overriding declaration for the appropriate parameter entity. For instance, the following declaration converts <note> to <annotation>:

```
<!ENTITY % n.note 'annotation'>
```

This declaration must be inserted in the file which has its name given by the entity *tei.extensions.ent*.

Two different cases of renaming can be identified. The first case involves replacing existing names with names that are otherwise unused in the TEI name space. (This can be easily checked by looking in the index of the Guidelines.) Such a modification is clean in that the new DTD would still accepts any document accepted by the publication DTD (given the appropriate renaming of elements). The new name cannot possibly conflict with the generic identifier of any other element, since there can be no other occurrences. To say this another way, the set of documents matching the new DTD is isomorphic to the set of documents matching the old DTD. The example given results in a clean modification because there is no element <annotation> specified in these Guidelines. It is also true that any document not using the renamed element which parses under the unmodified DTD will also parse under the modified DTD.

The second case involves introducing a name already used somewhere in a TEI tag set. This is unclean in that it changes what an existing generic identifier means. The name in question could not be declared by any tag set that is used in the document, as it is syntactically invalid to provide two declarations for the same element. The new generic identifier might occur in some TEI tag set not currently included in the DTD used to parse the document. For example, if in some setting the element <note> were assigned the new name <fs> (because, say, notes are used in some technical document to record functional specifications) there might be no immediate problem. If however it was later decided to add the feature structure analysis tag set into the DTD used to parse the document, though, a name clash would occur. There would also be problems in interchanging the resulting documents without confusion.

As a special case, consider translating all of the generic identifiers for all elements into some other language, L. It may be, for example, that the word for "paragraph" in language L begins with the letter "s" and that thus the paragraph element is renamed as <s>. By the definition just given, this would be an unclean modification because an element <s> already exists in the TEI DTD. However, this is clearly not a problem so long as all of the names are redefined at once and that no collisions occur in the new name space: that is, provided that the TEI element

<s> is renamed as some other string. This can be done by a total replacement of the file that contains the entity declarations for the names of the elements. This systematic replacement of names in the DTD must be followed by a systematic use (or replacement) of the new names in the document. To think about this in the terms used earlier, the set of documents matching the new DTD (with all names systematically changed in both the DTD and the documents) is isomorphic to the set of documents matching the original DTD with no names changed (in either the DTD or the documents).

The formal declarations of the parameter entities used for generic identifiers are contained in the file *teigis.ent*; since their names and replacement texts are fully predictable, these parameter entities are not individually documented in the reference section of these Guidelines. The parameter entity *tei.elementNames* is used to embed the file *teigis.ent* in the DTD. A re-declaration for this parameter entity may therefore be used to embed a different version of this file:

```
<!ENTITY % tei.elementNames system 'OTAgis.ent' >
```

If an element is renamed using the techniques described here, its declaration for the global TEIform attribute will be left undisturbed; the default value will therefore still be the standard TEI name for the element. TEI-aware application programs can thus process TEI-conformant documents which rename TEI elements, since by consulting the TEIform attribute value the application can learn the standard name for the element and process it accordingly.

In the normal course of events, this value of this attribute will never be specified in a TEI-conformant document; all occurrences will have the default value. In some special circumstances, it can be useful to specify a non-default value on some instances of an element; this allows application programs to process correctly a locally defined element which usually corresponds to one TEI element (which would be expressed by the default value) but sometimes to another TEI element (which would be expressed by explicit values attached to the element instance).

29.1.3 Class Extension

In 3.7.2 ('Classes Used in Content Models') on p. 54, the concept of a class of elements that can appear in the same kinds of structural locations in a document was introduced. An entity is associated with each class named by prefixing the string "m." (for "model") to the name of the class. For example, the value of the parameter entity *m.bibl* is a list of the members of the class *bibl*.

In the modifiable version of the TEI DTD, an additional entity is defined for each model class. This additional entity also takes the name of the class, this time prefixed by the string "x." (for "extension"). The default value of these *x-dot entities* is always the empty string. A reference to the corresponding x-dot entity is always included within the replacement string for each m-dot entity. This enables an encoder to add new members to a class simply by declaring a new value for its associated x-dot entity.

For example, the class *bibl* has the three members <bibl>, <biblFull>, and <biblStruct>. Its content-model entity is defined thus:

```
<!ENTITY % x.bibl '' >
<!ENTITY % m.bibl '%x.bibl bibl | biblFull | biblStruct' >
```

With the default value of the x-dot entity, this is the same as defining *m.bibl* with the replacement text `bibl | biblFull | biblStruct`.

An encoder can add an element to the class by providing a new declaration for the x-dot entity. For example, to add a new element called <myBib>, this definition would be used:

```
<!ENTITY % x.bibl 'myBibl |' >
```

Note that the specification of an x-dot entity must always end with the vertical bar character (for alternation). The definition would be inserted at the appropriate place in the file associated with the entity *tei.extensions.ent*. This changes the replacement text of *m.bibl* from its default value to `myBibl | bibl | biblFull | biblStruct`. If more than one element is to be added to a class, the x-dot entity for the class should be redefined as a list of the new generic identifiers, each one (*including the last*) followed by a vertical bar.

Class extension is always clean in that the set of documents matching the DTD containing the extended class contains all of the documents matching the original DTD. Class extension can

be by adding an existing element to a class, or by adding a new element (as described in the next section) and extending the class with the new element.

29.1.4 New content models

Encoders can modify the content models that specify what is contained in an element or set of elements defined by the TEI DTD, modify the attributes of existing elements, or add new elements to the DTD.

Content models or attributes for existing elements are modified in two stages. First, the existing definition of the element must be deleted in the manner described in the first section of this chapter. Second, a new definition of the element is given. This new definition must be inserted in the file associated with the entity *tei.extensions.dtd*.

For example, suppose that symbolic designations to be marked with the element <term> can always be associated with a particular source. While the content model of the publication version of the TEI DTD is acceptable, the attribute list needs to be extended. To perform this modification, the following steps must be taken. The declaration

```
<!ENTITY % term 'IGNORE' >
```

must be inserted into the file associated with the entity *tei.extensions.ent*. Then a new definition must be inserted into the file associated with the entity *tei.extensions.dtd*. In this example, the definition will be the same as that given in 6.3.4 ('Terms, Glosses, and Cited Words') on p. 130, save for the addition of a new attribute.

```
<!ELEMENT term          - -  (%phrase.seq;) >
<!ATTLIST term               %a.global;
          type               CDATA              #IMPLIED
          source             CDATA              #IMPLIED       >
```

New elements are defined by inserting their definitions into the file associated with the entity *tei.extensions.dtd*. To be usable, they must somehow be included in the model for some existing element. This can be done either by class extension (which can now be seen to be a restricted, special case of the process defined here) or by redefining the element(s) within which the new element is to be included.

The set of documents matched by the modified DTD and the set of documents matched by the original DTD may be related in several different ways. It is certainly possible that the former could properly include the latter or vice versa; either of these could be said to be clean modifications because the set to be matched has become strictly larger or strictly smaller.

It is also possible that the set of documents matched by the modified DTD is different from the set matched by the original DTD and they may either contain some common documents or have no documents in common; either of these is said to be an unclean modification.

Radical revision is possible. It would be possible to remodel so that the <teiHeader> is not required, or that it is required but the minimal components described in chapter 5 ('The TEI Header') on p. 77, are no longer required, or that no <text> element is required. In fact, the mechanism, if used in an extreme way, permits deletion of the entire set of TEI definitions and their replacement by an entirely different DTD! Such revisions would result in documents that are not TEI conformant in even the broadest sense, and it is not intended that encoders use the mechanism in this way.

29.2 Documenting the Modifications

When the modification mechanisms are used, their use must be documented. There are two ways in which information about the modifications is recorded.

The first record of the modifications is in the use of the extension files. The file associated with the entity *tei.extensions.ent* contains the specifications of the parameter entities that are redefined to accomplish the modifications. This file should be structured in such a way that readers can easily identify any modifications that have been made. The following structure is recommended.

```
<!--  The following elements are deleted     -->
<!--  The following elements are renamed     -->
<!--  The following classes are extended     -->
<!--  The following elements are revised   -->
```

The appropriate parameter entity specifications should be entered after each comment in this file. The order of the comments corresponds to the order of the discussion in this chapter (roughly, from simple to complex). Setting the appropriate entity to `IGNORE` for each revised element is done in the last section of the file.

The file associated with the entity *tei.extensions.dtd* should contain the DTD fragments for new and changed element definitions. The following structure is recommended.

```
<!--  The following declarations define new extensions    -->
<!--    (element and attlist specifications for new tags  -->
<!--    introduced in part 3 of the ent file go here)     -->
<!--  ...                                                 -->
<!--  The following declarations define revised tags    -->
<!--    (element and attlist specifications for tags      -->
<!--    mentioned in part 4 of the ent file go here       -->
<!--  ...                                                 -->
```

These files give an SGML parser sufficient information to implement the modifications and are also useful in providing human readers with some indication of the changes made in the TEI DTD. Full documentation of any additional or modified elements should also be provided, using the ancillary tag set described in chapter 27 ('Tag Set Documentation') on p. 601.

Chapter 30

Rules for Interchange

This chapter describes how interested parties can determine and agree on the proper format for the successful interchange of TEI-conformant documents over a given communications link, and how to translate from normal TEI form to the transmission format and back. It also includes recommendations for formats to be used when private arrangements cannot be made, in non-negotiated or "blind" interchange.

30.1 Negotiated Interchange

When the sender and receiver of a given text know each other's identity and can make appropriate special arrangements for the interchange of the text, the following procedures may be used to ensure the successful interchange of the text without information loss.

The sender and receiver together must first:

1. agree on whether to exchange the document in *TEI interchange form* or in some other form (e.g. *TEI local-processing form*)
2. identify the *communications link*: the method to be used to transmit and receive the document (transmission over a given network, physical transmission of a disk, tape, or other medium, etc.)
3. identify (by experimentation if necessary) the set of characters which can be transmitted successfully, without corruption, over the communications link; this is the *transmission character set*.
4. identify the set of characters present in the document which lie outside the transmission character set; this is the set of *non-transmissible characters*. For each character in this set, the local writing system declaration should identify an entity name or a transliteration into the transmission character set, which is to be used to transmit the character. The set of entities needed for this purpose is the *transmission entity set*.

The transmission character set is defined as the set of characters in the sender's system character set(s) which survive transmission and are properly recognized in the recipient's system character set. It is therefore by definition a subset of both the sender's and the recipient's system character sets. The bit patterns used to represent the characters may differ in the two systems (e.g. one may use ASCII, the other EBCDIC) if the communications link performs the proper translations.

Current network standards allow — indeed, require — gateway nodes to translate material passing through the gateway from one coded character set into another, when the networks joined by the gateway use different coded character sets. Since there is no universally satisfactory translation among all coded character sets in common use, the transmission character set will normally be the subset which is satisfactorily translated by the gateways encountered in transit between the sender and the receiver of the data.

When material is transmitted on a physical storage medium (e.g. disk or tape), then those exchanging documents have far greater control over the data. If both partners use

compatible systems, the transmission character set may be equivalent to their system character sets; otherwise, the transmission character set will include those characters which the recipient can successfully read into the local system character set from the media provided by the sender. For example, if a diskette created by an MS-DOS machine can be mailed to a Macintosh user and read directly by the recipient's system, then the transmission character set is likely to be ISO 646 IRV (equivalent to ANSI X3.4, or ASCII), which both machines have in common; if the recipient's disk-reading utilities are more sophisticated, however, then it may be possible to include some or all of the two machines' non-standard extended characters as well.

The mapping from non-transmissible characters to the transmission entity set may be derived from the writing system declarations in use by the sender and receiver.

After the transmission character set and entity character sets have been defined, the sender must prepare and transmit the document:

1. if the document is to be exchanged in TEI interchange format, translate it from the local-processing form into the TEI interchange form (e.g. by passing it through an SGML normalizer to supply omitted SGML tags and interpret all short-reference sequences)
2. pack the document for transmission by replacing every non-transmissible character with the appropriate transliteration or entity reference. At this point, every character in the document is represented either by a character in the transmission character set or by a reference to an entity in the transmission entity set; the document is in *TEI packed format*. If the transmission character set includes all the SGML delimiter characters, the document will be a conforming SGML document; otherwise, it may not be SGML-conformant but will be easily mappable to a conforming document
3. transmit the document over the agreed link

The translation of non-transmissible characters into the transmission entity set may be accomplished under automatic control, by software which reads the appropriate local writing system declarations, creates the necessary mapping tables, and packs the document for transmission.

Upon receipt, the receiver must:

1. unpack the document by expanding those entities which correspond to characters in the local system character set(s); at this point, the document should once more be a SGML-conformant, and should probably be validated to detect any problems in transmission
2. optionally, material transliterated in the document as transmitted may be translated into the local system character set(s) for more convenient display, or from the transmission entity set into a local transliteration scheme

It is strongly recommended that when documents are interchanged they be accompanied by any writing system declarations and feature system declarations which are applicable. In TEI-conformant interchange, it is required that documents be accompanied by any applicable tag-set documentation files.

30.2 Some Simple Examples

As a first simple example, consider a document containing English, French, and German, to be transmitted from an IBM-compatible personal computer to a Macintosh, over a long-distance network connection. Uploading test files from the PC to the sender's local network node, sending the file via the network, and downloading the document to the Macintosh, reveal (let us assume) that while all the characters of ISO 646 IRV survive intact, the accented characters of French and German do not survive transmission. In this case, the transmission character set is composed of all the characters of 7-bit ISO 646. The entity set required to handle the non-transmissible characters will include the following (assuming they actually occur in the document):

agrave
ccedille
eacute
egrave

ocirc
oelig
auml, Auml
ouml, Ouml
uuml, Uuml
szlig

In "packing" the file for transmission, the sender must replace the non-transmissible characters in the document with references to these entities. After this subsitution, the document is a conforming SGML document written entirely in the transmission character set, which can be sent over the communications link without any garbling or loss of information. Upon receipt, the recipient can replace the entity references with the specific coded characters used on the Macintosh to write French and German.

As a second example, consider the same document being transmitted from a VAX running VMS to an IBM mainframe running VM/CMS. Here, the accented characters might be represented on the VAX using the coded character set ISO 8859-1, but since ISO 8859-1 is not always supported, it may be more likely that entity references will be used instead. Let us assume that the network path between the two machines accepts Latin characters and digits, and most punctuation, but garbles square brackets, braces, the hash mark, and the pounds-sterling symbol. In this case, the accented characters require no special work by the sender, since they are already in a network-safe form. Square brackets, etc., must however be replaced by entity references to *lbr*, *rbr*, etc. After this is done, the document is no longer conformant SGML, since the square brackets used in certain markup declarations will not be recognized. (It is important, therefore, for validation to be performed before the square brackets are replaced by entity references.)

Upon receipt, the document may be translated into a valid SGML document by replacing all references to *lbr* and *rbr* with the appropriate square brackets, etc. If the local system supports one of the IBM code pages with support for French and German characters, then the entity references to those characters may be replaced by the characters in the system character set. More commonly, the entities for French and German characters will be left in place.

As a third example, consider a document containing Greek, as well as Latin, German, French, and English. If the sender's system has a full Greek character set, but the recipient's does not, then the Greek characters must all be replaced either by references to SGML entities or by transliterations into Latin characters (e.g. using the beta code transliteration developed for the Thesaurus Linguæ Græcæ). If the text is later transmitted to another system which does have a full Greek character set, the transliterated text or entity references may be translated, under control of the relevant writing system declarations, into the local Greek character set.

As a final example, consider a document written in Japanese, to be transmitted over a network within Japan, or over an international network to a recipient in Europe. Since networks within Japan transmit Japanese text without information loss, and common utilities may be used to recognize any of the existing coded character sets and translate into another, the transmission character set for interchange within Japan may be the same as the system character set. (If user-defined extensions are defined, in order to allow the encoding of kanji not present in the standard character sets, then these non-standard kanji may need to be replaced either by entity references or by "transliterations" into the standard character sets, and the description of the kanji themselves should accompany the documents in which they are used; the writing system declaration may be used for this purpose.)

When transmitting Japanese text outside Japan, the limitations of the networks at the time of transmission must be taken into account; it may be necessary to transliterate the text, or to replace non-transmissible characters with entity references.

30.3 Non-Negotiated Interchange

In some cases, no negotiation between interchange partners is possible, because they do not know each other's identities. Since it is impossible to discover an appropriate transmission entity set by experiment, such interchange requires the use of extremely conservative assumptions about the

frailties of network gateways.

Specifically, it is recommended that for non-negotiated interchange the following practices be adopted:

- The transmission character set should be the International Reference Version of ISO 646 (commonly known as ASCII). If the material is distributed on a physical medium, the material should actually be encoded in ISO 646; if the material is distributed over a network, the actual coded character set used at the point of origin is immaterial, but the data should still be restricted to the repertoire of ISO 646 (IRV).
- The transmission entity set should include only entities documented in ISO standard entity sets, published TEI writing system declarations, or writing system declarations made available with the material.
- All applicable writing system declarations should be distributed together with the material they describe; in non-negotiated interchange, all writing system declarations should assume ISO 646 (IRV) as the system character set.
- The SGML declaration distributed with the material should assume ISO 646 (IRV) as the system character set.
- If the receiver is not using ISO 646 IRV as the system character set, then the receiver (or some intervening network node) must translate from ISO 646 into the receiver's system character set. The receiver or the receiver's unpacking software is responsible for rewriting those parts of the SGML declaration which are dependent on the coded character set, and ensuring that they work properly on the receiving system.
- As transmitted, the document should be a valid SGML document.

By requiring the material to use only the characters of ISO 646 (IRV), and requiring the SGML declaration and accompanying writing system declarations to assume ISO 646 (IRV) as the system character set, these recommendations ensure that documents interchanged in this way will be directly usable on a great variety of systems; moreover, since ISO 646 is widely known and well documented, users of other systems will normally be able to adjust the documentation and data stream to their local systems without difficulty.

It should be noted that ISO 646 imposes a very restrictive and cumbersome encoding for researchers whose character sets have a large repertoire; it is strongly recommended, therefore, that arrangements for interchange of such materials involve explicitly negotiated interchange formats wherever possible.

The rules given here for non-negotiated interchange are not guaranteed to succeed, and negotiation of interchange formats is therefore *required*, if any of the following apply:

- The communications link between the interchange partners garbles or corrupts any characters of ISO 646 (IRV), (i.e., ISO 646 (IRV) is not a subset of the transmission character set).
- The communications link maps characters not in the repertoire of ISO 646 (IRV) onto characters in that repertoire.
- The document uses non-standard SGML syntax.

30.4 Notes for Implementors

The descriptions of document interchange in this chapter from time to time refer to software used to pack documents for interchange, or to unpack documents upon receipt. The descriptions do not characterize any specific existing software, but attempt to make clear how such software must work, in a general way. It is hoped that the descriptions will be useful to implementors of packing and unpacking software, but the full specification of such packing and unpacking software is beyond the scope of this chapter. All that can be attempted here is to describe some complications which may arise in the packing and unpacking of documents for interchange, of which implementors of such software should be aware.

- if the sender and receiver use different SGML syntaxes, various incompatibilities may be encountered which will require one partner or the other to modify the SGML syntax used

for the document, or the document itself, or both. In general, unless other arrangements are made, the responsibility for such modifications falls upon the receiver of the document.

- in particular, if the SGML syntax has been modified to expand the set of legal name characters (e.g. to allow characters with diacritic marks to occur in SGML names), then either the recipient must similarly modify the local SGML declaration, or the SGML names must be modified (by sender or receiver) to make them legal under the recipient's SGML declaration; name collisions must be carefully avoided when this is done.
- if the transmission character set does not include all characters with special meaning to SGML (name characters, delimiters, etc.), then although the packed document will be in a one-to-one relationship with a conforming SGML document, it will not itself be a conforming SGML document. In this case, final SGML validation must be done by the sender before packing, and the recipient cannot validate the received document before unpacking it.
- the proper packing of characters for transport may vary from language to language: in English text, a left square bracket may need to be packed with an entity reference to *lbr*, while the same bracket may have a different meaning, and thus a different entity replacement, in Greek text. In general, this means the packing should be done by an application with enough SGML awareness to detect element boundaries, read the value of the lang attribute from the start-tag or infer it from context, and adjust its actions appropriately. If only one WSD is in use in a given document and no language shifts are present, then both packing and unpacking may be done by a much simpler string-replacement algorithm.

Chapter 31

Multiple Hierarchies

At various points in these Guidelines, the discussion has mentioned the problems which arise when using SGML to encode textual features which do not take a strictly hierarchical form: features, that is, which do not necessarily nest within other features. This chapter provides an overview of the techniques defined in these Guidelines for handling such problems, and should be consulted when deciding how to deal with them.

The following examples illustrate the type of problem with which this chapter is concerned:

- in narrative, a speech by a character may begin in the middle of a paragraph and continue for several more paragraphs
- in a verse text, the encoder may need to tag both the formal structure of the verse (its stanzas and lines) and its syntactic structures (which sometimes nest within the metrical structure and sometimes cross metrical boundaries)
- in any kind of text, the encoder may wish to record the physical structure of volume, page, column, and line, as well as the formal or logical structure of chapters and paragraphs or acts and scenes, etc.
- in verse drama, the structure of acts, scenes, and speeches often conflicts with the metrical structure
- in any kind of text, an embedded text (e.g. a play within a play, or a song) may be interrupted by other matter; the encoder may wish to establish explicitly the logical unity of the embedded material (e.g. to identify the song as a single song, and to mark its internal formal structure)
- in a dictionary, different types of information (e.g. orthography, syllabification, and hyphenation) may be combined within a single notation; the encoder may wish both to preserve the presentation of the material in the source text and to disentangle the logically distinct pieces of information in the interests of more convenient processing of the lexical information

Many other examples might be given, but these should suffice to show the variety of applications where non-hierarchical or non-nesting information appears, and to illustrate the various methods for addressing the problem.

Non-nesting information poses fundamental problems for any encoding scheme, and it must be stated at the outset that no solution has yet been suggested which combines all the desirable attributes of formal simplicity, capacity to represent all occurring or imaginable kinds of structures, suitability for formal or mechanical validation, and clear identity with the notations needed for simpler cases (i.e. cases where the textual features do nest properly). The representation of non-hierarchical information is thus necessarily a matter of choices among alternatives, of tradeoffs between various sets of different advantages and disadvantages.

There are several methods used within these Guidelines to handle non-nesting information in SGML:

- *concur* : an optional feature of SGML which allows multiple hierarchies to be marked up concurrently in the same document
- *milestone elements*: empty elements which mark the boundaries between elements in a non-nesting structure
- *fragmentation* of an item: the division of what logically is a single element into two or more parts,

each of which nests properly within its context
- *virtual joins*: the recreation of a virtual element from fragments of text, possibly discontiguous or out of order
- redundant encoding of information in multiple forms

In the sections which follow, these techniques, their advantages, and their disadvantages, are briefly described, and instances of their use within these Guidelines are pointed out. The examples show various solutions to the problem of direct speech spanning several paragraphs in a narrative; the text in question takes the following form:

"

"The first thing that put us out was that advertisement. Spaulding, he came down into the office just this day eight weeks with this very paper in his hand, and he says:—"

""I wish to the Lord, Mr. Wilson, that I was a red-headed man.""

""Why that?" I asks.""

31.1 Concurrent Markup of Multiple Hierarchies

CONCUR allows us to mark up the document with many different hierarchical structures, but not to reorder the tree or to have different content in different views. Note that the restriction against having different content in different views is imposed not by SGML but by the TEI Interchange Format.

For example, if quotations are marked as part of a distinct markup stream given the name "QD", the outermost speech in our example need not be broken up into multiple elements:

```
<(QD)q who=Wilson> ...
<(TEI.2)p>The first thing that put us out was that
advertisement.  Spaulding, he came down into the
office just this day eight weeks with this very paper
in his hand, and he says:&mdash;</(TEI.2)p>
<(TEI.2)p><(QD)q who=Spaulding>I wish to the Lord, Mr.
Wilson, that I was a red-headed man.</(QD)q></(TEI.2)p>
<p><(QD)q who=Wilson>Why
that?</(QD)q> I asks.</(TEI.2)p>
 ... </(QD)q>
```

This method has the advantages of cleanly distinguishing among separate logical hierarchies in the text, using the same structures as non-concurrent markup and thus requiring no special conventions for use (as the other methods described in this chapter do). It has the disadvantage of using a cumbersome notation, which means it could most conveniently be used within an SGML-aware editing environment which masks the complexity of the notation from the user; unfortunately, CONCUR is an optional feature of SGML and is not supported by all SGML processors.

The major use of concurrent markup in the current version of these Guidelines is in the tag set for concurrent markup for pages, columns, and lines defined elsewhere in this chapter.

31.2 Boundary Marking with Milestone Elements

Milestones use empty elements to mark the beginnings and endings of regions of the text which have something in common; they work like COCOA tags. Examples in these Guidelines include the <milestone> element and the elements <pb>, <lb>, and <cb>.

For example, if quotations are marked using (user-defined) empty elements given the names "QB" and "QE", then no element contains the speeches and they need not be broken up into multiple elements at the paragraph breaks:

```
<qb who=Wilson> ...
<p>The first thing that put us out was that
```

```
advertisement.  Spaulding, he came down into the
office just this day eight weeks with this very paper
in his hand, and he says:&mdash;</p>
<p><qb who=Spaulding>I wish to the Lord, Mr.
Wilson, that I was a red-headed man.<qe></p>
<p><qb who=Wilson>Why that?<qe> I asks.</p>
 ... <qe>
```

This has the drawback that it is difficult to tell which <qe> corresponds with which <qb> without a complex processing of the text. One way to improve on this situation would be to use the linking attribute corresp discussed in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331 to associate the milestone indicating the end of a given speech with that indicating its start, as follows:

```
<qb who=Wilson id=W1> ...
<p>The first thing that put us out was that
advertisement.  Spaulding, he came down into the
office just this day eight weeks with this very paper
in his hand, and he says:&mdash;</p>
<p><qb who=Spaulding id=S1>I wish to the Lord, Mr.
Wilson, that I was a red-headed man.<qe corresp=S1></p>
<p><qb who=Wilson id=W2>Why that?<qe corresp=W2> I asks.</p>
 ... <qe corresp=W1>
```

This method has the advantage of simplicity; it provides all the information needed to reconstruct all the competing hierarchical views of the text. Many times, the only processing required for an element occurs at its start and end (or can easily be formulated to do so); this markup method handles those cases well. In other cases, however, this method incurs the disadvantage of cumbersome processing: since the elements of the analysis (e.g. the direct speech of Wilson) are not uniformly represented by nodes in the document tree, they must be reconstituted by software in an ad hoc fashion, which may be difficult and is likely to be error prone. Processing elements may often involve more than specified actions at the start and end of an element. Most important for some encoders, this method disguises the logical relationship between the beginning and the ending of each logical element, making it impossible for SGML parsers to provide the same kind of validation possible elsewhere in the encoding.

31.3 Fragmentation of Elements

Fragmentation breaks up what might be considered a single element into multiple smaller elements, in order to make it fit within the hierarchy. If a passage of direct discourse begins in the middle of one paragraph and continues for several more paragraphs, for example, one could encode the passage as a series of <q> elements. This has the effect that the document contains more <q> elements than it did before; to the extent that one might be interested in counting <q> elements, this is a drawback. To the extent that the element being broken up is used primarily to signal some characteristic, (e.g. that of being spoken by someone other than the narrator) rather than some countable object, this drawback is rather minor. Direct discourse is in fact so frequently interrupted by narrative irruptions, including but not limited to reporting clauses like "he said", that the number of <q> elements is unlikely to correspond precisely to the number of utterances, speaker turns, or any other observable unit of conversation. For that reason, some encoders prefer to solve the quotation-and-paragraph problem using fragmentation.

To tag our example with this method, the outermost speech (Wilson's) can be broken up to fit into the series of paragraphs, using the rend attribute to record the absence of closing quotation marks at the end of each paragraph. The inner speeches, being punctuated conventionally, need not carry rend values.

```
<p><q who=Wilson rend='pre ldquo'>The first thing that
put us out was that advertisement.  Spaulding, he came
down into the office just this day eight weeks with
```

```
this very paper in his hand, and he
says:&mdash;</q></p>
<p><q who=Wilson rend='pre ldquo'><q who=Spaulding>I
wish to the Lord, Mr. Wilson, that I was a red-headed
man.</q></q></p>
<p><q who=Wilson rend='pre ldquo'><q who=Wilson>Why
that?</q> I asks.</q></p>
```

Among the places where these Guidelines recommend fragmentation as a solution to the encoding of non-nesting information are the discussion of fragmentary verse lines, fragmentary stanzas, and fragmentary embedded texts in drama.

The advantages of this method are that it is simple, that at least one of the competing hierarchies can be processed normally, and that it makes the reconstitution of virtual units much easier, using the method described in the next section. Its disadvantages are that some units are not realized at all in the markup (here, the single long outermost speech of Wilson), and that automatic processing of these units is thus impossible when this method is used without further refinement.

31.4 Reconstitution of Virtual Elements

Virtual joins may be used to indicate objects in the text which would, for whatever reason, be difficult to mark using normal SGML syntax. In the TEI encoding scheme, virtual joins are most often expressed by the <join> element, the element, or the <fs> element. This technique covers all out-of-line annotation methods, which involve the construction, out of line, of a clean structure representing the interpretation, and the virtual join of the interpretation with the text fragment instantiating it.

The tagging of our example with this method is almost identical to that given in the preceding section, with the addition of <join> elements to indicate the component parts of the individual speeches which have been broken up to fit into the paragraph hierarchy:

```
<p><q id=qw1 who=Wilson rend='pre ldquo'>The first thing that
put us out was that advertisement.  Spaulding, he came
down into the office just this day eight weeks with
this very paper in his hand, and he
says:&mdash;</q></p>
<p><q id=qw2 who=Wilson rend='pre ldquo'><q who=Spaulding>I
wish to the Lord, Mr. Wilson, that I was a red-headed
man.</q></q></p>
<p><q id=qw3 who=Wilson rend='pre ldquo'><q who=Wilson>Why
that?</q> I asks.</q></p>

<!-- ... -->
<join targets='qw1 qw2 qw3' result='q'>
```

Alternatively, the next and prev attributes defined in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331 may be used to join the fragmentary quotations:

```
<p><q id=qw1 next=qw2
      who=Wilson rend='pre ldquo'>The first thing that
put us out was that advertisement.  Spaulding, he came
down into the office just this day eight weeks with
this very paper in his hand, and he
says:&mdash;</q></p>
<p><q id=qw2 next=qw3 prev=qw1
      who=Wilson rend='pre ldquo'><q who=Spaulding>I
wish to the Lord, Mr. Wilson, that I was a red-headed
man.</q></q></p>
<p><q id=qw3 prev=qw2
      who=Wilson rend='pre ldquo'><q who=Wilson>Why
that?</q> I asks.</q></p>
```

The major advantage of this method is that it allows all the hierarchies in the text to be handled explicitly, both the privileged one directly represented in the SGML and the alternate hierarchy which has been split up and rejoined. Its major disadvantages are that (like most of the other methods described here) it privileges one hierarchy over the others, and requires special processing to reconstitute the elements of the other hierarchies.

Instances of this markup method in these Guidelines include the part attribute on the <l>, <lg>, <seg>, and numbered-segment elements, the <join>, , and <fs> elements, and the global next and prev attributes available with the additional tag set for linking and alignment.

31.5 Multiple Encodings of the Same Information

In some cases, the simplest method of disentangling two conflicting hierarchical views of the same information is to encode it twice, each time capturing a single view. Thus, for example, a dictionary headword which gives in a single place the orthography, stress pattern, syllabification, and hyphenation for a word might be encoded several times: once with all the information in a single notation (as in the print dictionary), and once again for each separate piece of information — or at least, once more for the orthography, to speed up the common operation of searching for the article for a given headword in the electronic dictionary.

The out-of-line treatment of annotation in the feature structure notation (defined in chapter 16 ('Feature Structures') on p. 397) may also be considered to fall under this rubric.

The advantages of this method of markup are that each way of looking at the information is explicitly represented in the data, and may be processed in straightforward ways, without requiring complex methods of disentangling information relevant to one view from information relevant only to other views. It has the disadvantage of requiring more space and of introducing redundant information into the encoding, with the resulting risk that one view may be updated without corresponding changes being made to the others, resulting in inconsistencies within the document. Excessive use of redundancy may also make it difficult to reconstruct the exact form of the original source text.

31.6 Concurrent Markup for Pages and Lines

Where the main purpose for encoding alternative hierarchies in a text is to represent competing referencing schemes describing the same basic text, the CONCUR mechanism of SGML provides a very natural solution.

One common form of traditional reference system specifies the page and line, or page, column, and line of a passage as it appears in some standard edition. Such references may be specified using a concurrent markup hierarchy which divides the body of a text into pages and lines or into pages, columns, and lines. Volumes may also need to be identified. The document type name should be a short identifier for the edition cited. The following tags may be used:

<vol> marks the individual volumes of a reference edition.
<page> marks pages in a reference edition.
<col> contains one column of a multi-column reference edition.
<line> contains one line of a reference edition.

Page and line numbers for an edition by Lachmann, for example, might be specified thus:

```
<(La)page n=37> <!-- Text from Lachmann, p. 37 -->
                ...
<(La)line n=32> <!-- Text from Lachmann, p. 37, line 32 -->
<(La)line n=33> <!-- Text from Lachmann, p. 37, line 33 -->
<(La)line n=34> <!-- Text from Lachmann, p. 37, line 34 -->
<(La)page n=38> <!-- Text from Lachmann, p. 38 -->
<(La)line n=1>  <!-- Text from Lachmann, p. 38, line 1 -->
<(La)line n=2>  <!-- Text from Lachmann, p. 38, line 2 -->
```

```
<(La)line n=3>  <!-- Text from Lachmann, p. 38, line 3 -->

                <!-- etc. -->
<(La)page n=39> <!-- Text from Lachmann, p. 39 --> ...
<(La)line n=18> <!-- Text from Lachmann, p. 39, line 18 -->
<(La)line n=21> <!-- Text from Lachmann, p. 39, line 21 -->
                <!-- etc. -->
<(La)page n=40> <!-- Text from Lachmann, p. 40 --> ...
                <!-- etc. -->
<(La)page n=41> <!-- Text from Lachmann, p. 41 --> ...
                <!-- etc. -->
```

The markup shown above would be interleaved with the normal markup for the document. Since SGML requires tags in concurrent markup streams to be labeled with their document type, however, the "normal" markup would need to have the notation `TEI.2` inserted before each tag's generic identifier. The combined markup might look something like this:

```
<(TEI.2)TEI.2>
<(TEI.2)TEI.Header> ... </(TEI.2)TEI.Header>
<(TEI.2)text> ...
<(TEI.2)div0><(TEI.2)head> ... </(TEI.2)head>
...
<(TEI.2)div1>
<(TEI.2)div2> ...
<(La)page n=37> <!-- Text from Lachmann, p. 37 -->
                ...
<(La)line n=32> <!-- Text from Lachmann, p. 37, line 32 -->
<(La)line n=33> <!-- Text from Lachmann, p. 37, line 33 -->
<(La)line n=34> <!-- Text from Lachmann, p. 37, line 34 -->
<(La)page n=38> <!-- Text from Lachmann, p. 38 -->
<(La)line n=1>  <!-- Text from Lachmann, p. 38, line 1 -->
<(La)line n=2>  <!-- Text from Lachmann, p. 38, line 2 -->
<(La)line n=3>  <!-- Text from Lachmann, p. 38, line 3 -->
</(TEI.2)div2>
<(TEI.2)div2>
<(La)line n=4>  <!-- Text from Lachmann, p. 38, line 4 -->
                <!-- etc. -->
</(TEI.2)div2>
</(TEI.2)div1>
<(TEI.2)div1><head>... </head>
                ...
<!-- Text from Lachmann, p. 39 --> ...
<(La)page n=39> <!-- Text from Lachmann, p. 39 --> ...
<(La)line n=18> <!-- Text from Lachmann, p. 39, line 18 -->
<(La)line n=19> <!-- Text from Lachmann, p. 39, line 19 -->
<(La)line n=20> <!-- Text from Lachmann, p. 39, line 20 -->
<(La)line n=21> <!-- Text from Lachmann, p. 39, line 21 -->
                ... <!-- etc. -->
<(TEI.2)div1>
<(TEI.2)div2>
</(TEI.2)div2>
<!-- etc. -->
<!-- Text from Lachmann, p. 40 --> ...
<!-- etc. -->
</(TEI.2)text>
</(TEI.2)TEI.2>
```

The following SGML declarations give the formal specification for the standard pre-defined document type designed for recording page and line numbers of a reference edition in a concurrent markup stream.

```
<!-- 31.6: Concurrent Document Type for Page and Line         -->
<!-- References                                               -->
```

```
<!ENTITY % version 'ref'                                          >
<!ELEMENT %version      - -  (#PCDATA | page | vol)*              >
<!ATTLIST %version           %a.global;                           >
<!ELEMENT vol           - -  (#PCDATA | page)*                    >
<!ATTLIST vol                %a.global;                           >
<!ELEMENT page          - O  (#PCDATA | line | col)*              >
<!ATTLIST page               %a.global;                           >
<!ELEMENT col           - O  (#PCDATA | line)*                    >
<!ATTLIST col                %a.global;                           >
<!ELEMENT line          - -  (#PCDATA)                            >
<!ATTLIST line               %a.global;                           >
```

This concurrent hierarchy is enabled as shown below: after the document type declaration for the TEI.2 document type, the document should contain the sequence of lines:

```
<!DOCTYPE La system 'teipl2.dtd' [
<!ENTITY % version "La" >
]>
```

which call the document type for page and line references and give it the name "La". If page and line numbers from more than one standard edition are to be marked, then the relevant lines may be repeated, each time using a different value for the document type and entity definition (where the example has "La"). For example, to show page and line numbers from the editions of Lachmann (La), Kraus (Kr), and Moser/Tervooren (MT) at the same time, one might use declarations like the following:

```
<!DOCTYPE La system 'teipl2.dtd' [
     <!ENTITY % version "La" >
]>
<!DOCTYPE Kr system 'teipl2.dtd' [
     <!ENTITY % version "Kr" >
]>
<!DOCTYPE MT system 'teipl2.dtd' [
     <!ENTITY % version "MT" >
]>
```

To document a referencing system of this kind the TEI header, a formal declaration should be provided in the <refsDecl> element described in section 5.3.5 ('The Reference System Declaration ') on p. 100. For the above example, a declaration such as the following would be appropriate:

```
<refsDecl n=La>
    <step dtd=La gi='page' att='n'>
    <step dtd=La gi='line' att='n'>
</refsDecl>
<refsDecl n=Kr>
    <step dtd=Kr gi='page' att='n'>
    <step dtd=Kr gi='line' att='n'>
</refsDecl>
<refsDecl n=MT>
    <step dtd=MT gi='page' att='n'>
    <step dtd=MT gi='line' att='n'>
</refsDecl>
```

Hierarchies similar to that defined above can be provided for most common hierarchical reference systems. Hierarchies such as act / scene / line, for conventional dramatic structure, book / canto / stanza / line, for longer verse texts, or book / poem / stanza / line, for collections of verse, may be readily expressed with concurrent SGML markup. Since these hierarchical structures can readily be represented using the base tag sets described in part III of these Guidelines, however, reference systems with such structures may most readily be expressed using the n or id attributes, as described above in section 6.9.1 ('Using the ID and N Attributes') on p. 156.

Any text with an idiosyncratic standard reference system will require its own dtd, so that appropriately named tags can be created for the reference units. Such dtds may follow the pattern

of those described in the preceding section; they should also be documented in an auxiliary tag set description file, using the tags described in chapter 27 (‘Tag Set Documentation’) on p. 601.

Chapter 32

Algorithm for Recognizing Canonical References

When a canonical reference is to be automatically processed according to this method, the following occurs:

1. The reference is analysed into a series of component targets, using the delim and length attributes on the <step> elements within the relevant <refsDecl>, as described below. The target from the first <step> element is made available for use in the subsequent pointer processing as %1, that from the second as %2, and so on. The number of targets N must be noted, as it will be used in later steps of the algorithm. (Targets with numbers greater than N may be referenced, but they will be null strings.)
2. Starting at the root of the tree (i.e. the <text> element, for a TEI-conformant document), a search is made following the specifications of the pointer in the first <step> element: its from and to attributes are processed exactly as would be done with an <xptr> element.
3. If there are more targets to be located, the search continues by following the pointer specifications in the next <step> element, using as location source the span located by the previous <step>. Except for this special location source, the processing is still identical to that done for an <xptr> element. (Note that this does not prevent an expansion of the location source at any step, either by using keywords such as previous which search outside the location source, or by an explicit return to root.)
4. When N <step> elements have been processed, the search is complete. The final result of the reference is a point or span of text, as with extended pointers.

Note that there is no backtracking or other attempt at recovery if a pointer fails. It is instead possible to design the pointers so that backtracking is not necessary: see the final example below. When analyzing a reference into component targets, the following procedure is adopted for each <step> element that is used:

1. If only the length attribute is specified, exactly that number of characters is taken from the reference. Entity references are resolved before characters are counted. (This implies that references containing entity references may behave differently on different systems.)
2. If only the delim attribute was specified, every character up to the next occurrence of the specified delimiter is taken from the reference, and the delimiter itself is removed.
3. If both length and delim attributes are specified, a test is made for the presence of the delimiter in the reference string immediately following the specified number of characters; if this test fails, the reference fails.
4. If neither length nor delim attribute is specified, the remainder of the reference string is taken. This should happen only for the last <step> in a <refsDecl>.
5. The number of components resulting from this procedure must not exceed the number of steps in the associated declaration, but may be less than it. (For example, a long poem might be divided into cantos and lines, but a reference can point to either a line in a canto or to a whole canto; a reference to a whole canto would not require the <step> for the line number.)

Here is an example of how a reference system for an encoding of the Bible could be specified:

```
<refsDecl>
 <step refunit="book" delim=" "
       from="CHILD (1 DIV N %1)"     to="DITTO">
 <step refunit="chapter" delim=":"
       from="CHILD (1 DIV N %2)"     to="DITTO">
 <step refunit="verse" delim=""
       from="CHILD (1 DIV N %3)"     to="DITTO">
</refs.decl>
```

With this reference declaration, a canonical reference of the form "Matt 5:7" is processed by first searching for the subelement of the <text> element with an n attribute having value Matt; then searching within that element for a subelement with an n attribute having value 5; and finally for a further nested element numbered 7. This example assumes that the unnumbered elements nevertheless follow a predictable hierarchy: the first level is always for books, the second for chapters, and the third for verses. The following reference declaration would allow intermediate elements of any sort, because it would search at each step not only for the right n attribute but also for a type attribute identifying the structural type of the division:

```
<refsDecl>
 <step refunit="book" delim=" "
       from="DESCENDANT (1 DIV N %1 TYPE BOOK)"     to="DITTO">
 <step refunit="chapter" delim=":"
       from="DESCENDANT (1 DIV N %2 TYPE CHAPTER)"  to="DITTO">
 <step refunit=verse delim=""
       from="DESCENDANT (1 DIV N %3 TYPE VERSE)"    to="DITTO">
</refs.decl>
```

Other reference systems depend on markers such as page and line numbers which do not correspond to structural divisions of the text. These will typically be marked in the text by milestone elements which identify single points in the text, rather than by structural elements which contain the portion of the text to be located. It is then necessary to construct extended pointers in the reference declaration that can locate both the start and the end of any segment. Here is a reference declaration for a work whose reference system consists of page and line numbers: for example, "93.3".

```
<refsDecl>
 <step refunit=page delim="."
       from="FOLLOWING (1 PB N %1)"
       to="FOLLOWING (1 PB)">
 <step refunit=line delim=""
       from="FOLLOWING (1 LB N %2)"
       to="FOLLOWING (1 LB)">
</refs.decl>
```

To locate the specified page, the application must first search for the first <pb> element with n equal to 93. It must then find the end of that page, which it does by searching for the next <pb> element after that for the start of page 93; this should mark the start of page 94. A similar procedure is used within the page to find the <lb> elements that mark the start and end of the desired line. A reference system may combine elements of the last two approaches: a reference system based on line numbers is normally used for early English plays, but because such plays often combine prose and verse the line numbers sometimes refer to structural elements (for verse passages) and sometimes to arbitrary typographical boundaries (for prose). One could simply fill the entire text with milestones, even in the verse passages which do not require them; but a reference declaration can also be constructed which requires no superfluous elements. Here is such a declaration for a collection of plays, and for canonical references of the form "Changeling 1.2.44".

```
<refsDecl>
 <step refunit=work delim=" "
       from="DESCENDANT (1 TEXT N %1)"
       to="DITTO">
```

```
 <step refunit=act delim="."
       from="CHILD (1 DIV1 N %2)"
       to="DITTO">
 <step refunit=scene delim="."
       from="CHILD (1 DIV2 N %3)"
       to="DITTO">
 <step refunit=line delim=""
       from="FOLLOWING (1 (L|LB) N %4)"
       to="FOLLOWING (1 (L|LB))">
</refs.decl>
```

Instead of using the CHILD or DESCENDANT keywords to locate the <l> elements in verse passages, we use FOLLOWING, which works to locate <lb> elements as well. The algorithm also allows ambiguity in the reference, as a substitute for requiring backtracking in the processing. Consider the following reference declaration for a text containing several works:

```
<refsDecl>
 <step refunit=work delim=" "
       from="DESCENDANT (ALL TEXT N %1)" to="DITTO">
 <step refunit=book delim="."
       from="CHILD (1 DIV1 N %2)"        to="DITTO">
 <step refunit=poem delim="."
       from="CHILD (1 DIV2 N %3)"        to="DITTO">
 <step refunit=line delim=""
       from="CHILD (1 L N %4)"           to="DITTO">
</refs.decl>
```

Given the canonical reference `Amores I.2`, the application will first search for all <text> elements whose `n` attribute has the value `Amores`. This creates a location source for the second step consisting of one or more <text> elements, possibly discontiguous; at the second step, all the <div1> elements numbered `I` within those <text> elements are selected. At the third, the contained <div2> elements numbered `2` are selected, and the search ends because the reference string is exhausted: it points to a whole poem, not to a single line. This reference declaration is designed for use with a text that contains several works, some of which might have the same name, so that at the first step it is not adequate to search merely for the first <text> called `Amores`. It may be that several works called `Amores` exist but only one has a book numbered `I`, so that at the second step the location source narrows down to the one work desired. In other approaches to this problem, the works called `Amores` would be treated one at a time, and if one turned out not to be the one desired backtracking to previous steps of the search would be required; this reference-processing method uses composite location sources instead to eliminate the need for backtracking.

Part VII

Alphabetical Reference List of Tags and Attributes

Chapter 33

Element Classes

Class Name: addrPart (i.e. address part) addrPart

Description: defines a group of elements which may constitute a postal or other form of address.

Attributes: [None: global and inherited attributes only.]

Part: base tag set for common core features

Member of classes:

DTD file: teiclas2.ent

Occurs within: address

Members of: postBox postCode street

```
<!ENTITY % m.addrPart '
  %x.addrPart street | postCode | postBox'                    >
```

Discussed in: 6.4.2 ('Addresses') on p. 134

Class Name: agent (i.e. agent (individual or corporate body)) agent

Description: defines a group of elements which contain names of individuals or corporate bodies.

Attributes: [None: global and inherited attributes only.]

Remarks:
This class is used in the <resp> element, to allow a statement of responsibility to apply to an individual or a body.

Part: base tag set for common core features

Member of classes:

DTD file: teiclas2.ent

Occurs within: [none]

Members of: name

```
<!ENTITY % m.agent '
  %x.agent name'                                              >
```

Discussed in: 6.4 ('Names, Numbers, Dates, Abbreviations, and Addresses') on p. 132

Class Name: analysis analysis

Description: default declaration for class *analysis*: when the additional tag set for simple analysis is not selected, no attributes are defined for this class.

Attributes: [None: global and inherited attributes only.]

Part: additional tag set for analysis and interpretation

Member of classes:

DTD file: teiclas2.ent

Occurs within: [none]

Members of:

```
<!ENTITY % a.analysis ''                                      >
```

Discussed in: 3.5 ('Global Attributes') on p. 42; 15 ('Simple Analytic Mechanisms') on p. 381

analysis

Class Name: analysis

Description: defines global attributes for associating specific analyses or interpretations with appropriate portions of a text, for use when the additional tag set for simple analysis is selected.

Attributes:

ana (i.e. analysis) indicates one or more elements containing interpretations of the element on which the ana attribute appears.
Data type: IDREFS
Value: The SGML identifier of one or more interpretive elements (usually <fs> or <interp>).
Default: #IMPLIED
When multiple values are given, they may reflect either multiple divergent interpretations of an ambiguous text, or multiple mutually consistent interpretations of the same passage in different contexts.

Part: additional tag set for analysis and interpretation
Member of classes:
DTD file: teiana2.ent
Occurs within: [none]
Members of: *global* ()

```
<!ENTITY % a.analysis '
          ana                 IDREFS              #IMPLIED'      >
```

Discussed in: 15.3 ('Spans and Interpretations') on p. 387

baseStandard

Class Name: baseStandard (i.e. base-component standard)

Description: groups elements in a writing system which refer to some public or private standard as part of the basis for the writing system declaration

Attributes:

name gives the normal citation form for the standard being referred to.
Data type: CDATA
Value: For national and international standards, the value should be the normal citation form for the standard; for public entity sets, it should be the standard public entity text; for private character sets, WSDs, and entity sets, it is recommended that the form of SGML *public identifiers* be used.
Default: #REQUIRED
Example:

```
<codedCharSet name='ANSI X3.4' authority='national'>
<codedCharSet name='ISO 646: 1991' authority='ISO'>
<entitySet    name='ISO 8879:1986//ENTITIES Added Latin 1//EN'
              authority='ISO'>
<baseWsd      name='-//TEI P2: 1993//WSD ISO 8859-1//EN'
              authority='TEI'>
```

authority indicates the authority responsible for issuing the standard being referred to: the TEI, the International Organization for Standardization (ISO), a national body, or a private body.
Data type: (TEI | ISO | NATIONAL | PRIVATE | NONE)
Sample values include:

tei the base writing system declaration is a standard WSD issued by the Text Encoding Initiative
iso the character set or entity set was issued by ISO
national the character set or entity set was issued by a national standards body
private the writing system declaration, character set, or entity set was issued publicly by a private organization or project
none the writing system declaration, character set, or entity set has not been publicly issued by any organization; it is specific to an individual text or project

Default: #REQUIRED

Remarks:

Part: auxiliary tag set for writing system declarations
Member of classes:
DTD file: teiwsd2
Occurs within: [none]
Members of: baseWsd codedCharSet entitySet

```
<!ENTITY % a.baseStandard '
          name               CDATA               #REQUIRED
          authority          (tei | iso | national | private |
                             none)               #REQUIRED'     >
```

Discussed in: 25.4.1 ('Base Components of the WSD') on p. 576

Class Name: bibl bibl

Description: bibliographic elements.
Attributes: [None: global and inherited attributes only.]

Remarks:

Part: base tag set for common core features
Member of classes: common, inter
DTD file: teiclas2.ent
Occurs within: add admin argument body camera caption case castList cell cit colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref
Members of: bibl biblFull biblStruct

```
<!ENTITY % m.bibl '
  %x.bibl biblStruct | biblFull | bibl'                          >
```

Discussed in: 6.10 ('Bibliographic Citations and References') on p. 162

Class Name: biblPart (i.e. bibliographic citation part) biblPart

Description: elements which can appear within bibliographic citations, but cannot appear freely in running text outside bibliographic citations.
Attributes: [None: global and inherited attributes only.]

Remarks:
This class is used in defining the content model of <bibl>.

Part: base tag set for common core features
Member of classes:
DTD file: teiclas2.ent
Occurs within: bibl
Members of: analytic author biblScope edition editor extent idno imprint monogr note publisher pubPlace respStmt series

```
<!ENTITY % m.biblPart '
  %x.biblPart series | respStmt | pubPlace | publisher | note |
  monogr | imprint | idno | extent | editor | edition |
  biblScope | author | analytic'                                 >
```

Discussed in: 6.10 ('Bibliographic Citations and References') on p. 162

Class Name: binary (i.e. binary feature-structure values) binary

Description: elements which express binary values in feature structures.
Attributes: [None: global and inherited attributes only.]
Part: additional tag set for feature structures

Member of classes: singleVal [and indirectly also:] featureVal
DTD file: teifsd2
Occurs within: if vDefault vRange
Members of: minus plus

```
<!ENTITY % m.binary '
  %x.binary plus | minus'                                    >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

boolean

Class Name: boolean (i.e. Boolean values)
Description: elements which express Boolean values in feature structures.
Attributes: [None: global and inherited attributes only.]
Part: additional tag set for feature structures
Member of classes: singleVal [and indirectly also:] featureVal
DTD file: teifsd2
Occurs within: if vDefault vRange
Members of: any none

```
<!ENTITY % m.boolean '
  %x.boolean none | any'                                     >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

chunk

Class Name: chunk
Description: includes all elements which can occur between, but not within, paragraphs and other chunks.
Attributes: [None: global and inherited attributes only.]

Remarks:
This element class does not contain all those elements which can appear between chunks: the class *inter* contains a set of elements which can appear either within or between chunks. Unlike elements of that class, chunks cannot occur within chunks. In prose, this means the elements in this class can appear between but not within paragraphs.

Part: base tag set for common core features
Member of classes: common
DTD file: teiclas2.ent
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage view
Members of: eTree graph l lg p sp tree

```
<!ENTITY % m.chunk '
  %x.chunk tree | sp | p | lg | l | graph | eTree'             >
```

Discussed in: 3.7 ('Element Classes') on p. 51

common

Class Name: common
Description: common chunk- and inter-level elements.
Attributes: [None: global and inherited attributes only.]

Remarks:
This class defines the set of chunk- and inter-level elements available in all bases; it is used in defining the standard models *chunk.seq* and *specialPara* in the general and mixed bases.

Part: base tag set for common core features
Member of classes:
DTD file: teiclas2.ent
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage view
Members of: stage *notes* (note) *lists* (label list listBibl) *hqinter* (cit q quote) *chunk* (eTree graph l lg p sp tree) *bibl* (bibl biblFull biblStruct)

```
<!ENTITY % m.common '
  %x.common %m.notes | %m.lists | %m.hqinter | %m.chunk |
  %m.bibl | stage'                                          >
```

Discussed in: 3.7 (‘Element Classes’) on p. 51

Class Name: comp.dictionaries comp.dictionaries

Description: component-level elements unique to the base tag set for dictionaries.
Attributes: [None: global and inherited attributes only.]

Remarks:

Part: base tag set for terminological data
Member of classes:
DTD file: teidict2.ent
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage view
Members of: entry entryFree superentry

```
<!ENTITY % m.comp.dictionaries '
  %x.comp.dictionaries superentry | entryFree | entry'          >
```

Discussed in: 12.1 (‘Dictionary Body and Overall Structure’) on p. 270; 3.7 (‘Element Classes’) on p. 51

Class Name: comp.drama comp.drama

Description: component-level elements specific to performance texts.
Attributes: [None: global and inherited attributes only.]
Part: base tag set for performance texts
Member of classes:
DTD file: teidram2.ent
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage view
Members of: castList *stageDirection* (camera caption move sound tech view)

```
<!ENTITY % m.comp.drama '
  %x.comp.drama %m.stageDirection | castList'                  >
```

Discussed in: 10 (‘Base Tag Set for Drama’) on p. 227

Class Name: comp.spoken comp.spoken

Description: elements appearing in spoken texts only.
Attributes: [None: global and inherited attributes only.]

Remarks:

Part: base tag set for spoken materials
Member of classes:
DTD file: teispok2.ent
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage u view
Members of: event kinesic pause shift u vocal writing

```
<!ENTITY % m.comp.spoken '
  %x.comp.spoken writing | vocal | u | shift | pause | kinesic |
  event'                                                       >
```

Discussed in: 11.1 (‘General Considerations and Overview’) on p. 250

Class Name: comp.terminology comp.terminology

Description: component-level elements unique to the base tag set for terminological data.
Attributes: [None: global and inherited attributes only.]

Remarks:

Part: base tag set for terminological data
Member of classes:
DTD file: teiterm2.ent
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage view

Members of: termEntry

```
<!ENTITY % m.comp.terminology '
  %x.comp.terminology termEntry'                                    >
```

Discussed in: 13.3 ('Basic Structure of the Terminological Entry') on p. 316; 3.7 ('Element Classes') on p. 51

comp.verse

Class Name: comp.verse

Description: component level elements unique to the base tag set for verse.
Attributes: [None: global and inherited attributes only.]
Remarks:
Part: base tag set for verse texts
Member of classes:
DTD file: teivers2.ent
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage view
Members of: lg1

```
<!ENTITY % m.comp.verse '
  %x.comp.verse lg1'                                                >
```

Discussed in: 9.2 ('Structural Divisions of Verse Texts') on p. 214; 3.7 ('Element Classes') on p. 51

complexVal

Class Name: complexVal (i.e. complex values)

Description: elements which express complex feature values in feature structures.
Attributes: [None: global and inherited attributes only.]
Part: additional tag set for feature structures
Member of classes: featureVal
DTD file: teifsd2
Occurs within: if vDefault vRange
Members of: alt fs vAlt

```
<!ENTITY % m.complexVal '
  %x.complexVal vAlt | fs | alt'                                    >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

data

Class Name: data

Description: phrase-level elements containing names, dates, numbers, measures, and similar data.
Attributes: [None: global and inherited attributes only.]
Remarks:
Part: base tag set for common core features
Member of classes: phrase
DTD file: teiclas2.ent
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
Members of: abbr address date dateRange dateStruct expan lang measure name num rs time timeRange timeStruct

```
<!ENTITY % m.data '
  %x.data timeStruct | timeRange | time | rs | num | name |
  measure | lang | expan | dateStruct | dateRange | date |
  address | abbr'                                                >
```

Discussed in: 6.4 ('Names, Numbers, Dates, Abbreviations, and Addresses') on p. 132

Class Name: date (i.e. dates and date ranges) date

Description: defines a class of elements containing date specifications.
Attributes: [None: global and inherited attributes only.]

Remarks:
This class allows certain content models to allow either a single date or a date-range element.

Part: base tag set for common core features
Member of classes:
DTD file: teiclas2.ent
Occurs within: [none]
Members of: date dateRange dateStruct

```
<!ENTITY % m.date '
  %x.date dateStruct | dateRange | date'                        >
```

Discussed in: 6.4 ('Names, Numbers, Dates, Abbreviations, and Addresses') on p. 132

Class Name: declarable declarable

Description: elements which may be independently selected (using the special purpose decls attribute) from a candidate list of declarations within a TEI header.
Attributes:

default indicates whether or not this element is selected by default when its parent is selected.
Data type: (YES | NO)
Sample values include:

YES This element is selected if its parent is selected

NO This element can only be selected explicitly, unless it is the only one of its kind, in which case it is selected if its parent is selected.

Default: NO

Remarks:
The rules governing the association of declarable elements with individual parts of a TEI text are fully defined in chapter 23.3 ('Associating Contextual Information with a Text') on p. 550. Only one element of a particular type may carry the value default=yes.

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teiclas2.ent
Occurs within: [none]
Members of: bibl biblFull biblStruct broadcast correction editorialDecl equipment hyphenation interpretation langUsage listBibl metDecl normalization particDesc projectDesc quotation recording samplingDecl scriptStmt segmentation settingDesc sourceDesc stdVals textClass textDesc

```
<!ENTITY % a.declarable '
          default            (YES | NO)          NO'           >
```

Discussed in: 23.3 ('Associating Contextual Information with a Text') on p. 550

Class Name: declaring declaring

Description: groups elements which may be independently associated with a particular declarable element within the header, thus overriding the inherited default for that element.
Attributes:

decls identifies one or more *declarable elements* within the header, which are understood to apply to the element bearing this attribute and its content.
Data type: IDREFS
Value: must identify a set of declarable elements of different types.
Default: #IMPLIED

Remarks:
The rules governing the association of declarable elements with individual parts of a TEI text are fully defined in chapter 23.3 ('Associating Contextual Information with a Text') on p. 550.

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: `teiclas2.ent`
Occurs within: [none]
Members of: back body div div0 div1 div2 div3 div4 div5 div6 div7 front group text

```
<!ENTITY % a.declaring '
          decls              IDREFS              #IMPLIED'      >
```

Discussed in: 23.3 ('Associating Contextual Information with a Text') on p. 550

demographic

Class Name: demographic

Description: elements describing demographic characteristics of the participants in a linguistic interaction.
Attributes: [None: global and inherited attributes only.]

Remarks:

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: `teicorp2`
Occurs within: person personGrp
Members of: birth education firstLang occupation persName residence socecStatus

```
<!ENTITY % m.demographic '
  %x.demographic socecStatus | residence | persName | occupation
  | firstLang | education | birth'                                  >
```

Discussed in: 23.2.2 ('The Participants Description') on p. 545

dictionaries

Class Name: dictionaries

Description: defines global attributes available on elements in the base tag set for dictionaries.
Attributes:

expand gives an expanded form of information presented more concisely in the dictionary
Data type: CDATA
Value: any string of characters
Default: #IMPLIED

norm gives a normalized form of information given by the source text in a non-normalized form
Data type: CDATA
Value: any string of characters
Default: #IMPLIED

split gives the list of split values for a merged form
Data type: CDATA
Value: any string of characters
Default: #IMPLIED

value gives a value which lacks any realization in the printed source text.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED

orig (i.e. original) gives the original string or is the empty string when the element does not appear in the source text.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED

mergedin gives a reference to another element, where the original appears as a merged form.
Data type: IDREF
Value: the SGML identifier of any element
Default: #IMPLIED

opt (i.e. optional) indicates whether the element is optional or not
Data type: (Y | N)
Value: any string of characters
Default: N
Example:

```
<form>
  <orth id=o1 next=o2>thyr</orth>
  <orth id=o2 prev=o1 next=o3 opt=y>&eacute;</orth>
  <orth id=o3 prev=o2>ostimuline</orth>

  <pron id=p1 next=p2>tiR</pron>
  <pron id=p2 prev=p1 next=p3 opt=y>e</pron>
  <pron id=p3 prev=p2>ostimylin</pron>
</form>
```

Part: base tag set for printed dictionaries
Member of classes:
DTD file: teidict2.ent
Occurs within: [none]
Members of: case colloc def eg entryFree etym form gen gram gramGrp hom hyph itype lang lbl mood number orth oRef oVar per pos pron pRef pVar re sense subc syll tns tr trans usg xr

```
<!ENTITY % a.dictionaries '
          expand              CDATA               #IMPLIED
          norm                CDATA               #IMPLIED
          split               CDATA               #IMPLIED
          value               CDATA               #IMPLIED
          orig                CDATA               #IMPLIED
          mergedin            IDREF               #IMPLIED
          opt                 (y | n)             n'             >
```

Discussed in: 12.2 ('The Structure of Dictionary Entries') on p. 274

Class Name: dictionaries (i.e. attributes for dictionary elements.) dictionaries

Description: default declaration for class *dictionaries*: when the base tag set for dictionaries is not selected, no attributes are defined for this class.
Attributes: [None: global and inherited attributes only.]
Part: base tag set for terminological data
Member of classes:
DTD file: teite2f
Occurs within: [none]
Members of:

```
<!ENTITY % a.dictionaries ''                                    >
```

Discussed in: 13.4.2 ('DTD Fragment for Flat Style') on p. 322

Class Name: dictionaryParts (i.e. dictionary parts) dictionaryParts

Description: groups all elements defined specifically for dictionaries.
Attributes: [None: global and inherited attributes only.]

Remarks:
(optional)

Part: base tag set for printed dictionaries
Member of classes:
DTD file: teidict2.ent
Occurs within: trans [May appear anywhere within: eg entryFree eLeaf]
Members of: case colloc def eg etym form gen gramGrp hom hyph itype lbl mood number orth per pos pron re sense stress subc superentry syll tns tr trans usg xr

```
<!ENTITY % m.dictionaryParts '
  %x.dictionaryParts xr | usg | trans | tr | tns | syll |
  superentry | subc | stress | sense | re | pron | pos | per |
  orth | number | mood | lbl | itype | hyph | hom | gramGrp |
  gen | form | etym | eg | def | colloc | case'                >
```

Discussed in: 12.1 ('Dictionary Body and Overall Structure') on p. 270

dictionaryTopLevel

Class Name: dictionaryTopLevel (i.e. dictionary top-level elements)
Description: groups elements which can appear at the "top level" in dictionary entries.
Attributes: [None: global and inherited attributes only.]
Part: base tag set for printed dictionaries
Member of classes:
DTD file: teidict2.ent
Occurs within: entry hom re sense
Members of: def eg etym form gramGrp note re trans usg xr

```
<!ENTITY % m.dictionaryTopLevel '
  %x.dictionaryTopLevel xr | usg | trans | re | note | gramGrp |
  form | etym | eg | def'                                       >
```

Discussed in: 12.2 ('The Structure of Dictionary Entries') on p. 274

divbot

Class Name: divbot (i.e. Bottom-of-division elements)
Description: groups elements which can occur at the end of a text division; for example, trailer, byline, etc.
Attributes: [None: global and inherited attributes only.]
Part: base tag set for common core features
Member of classes:
DTD file: teistr2
Occurs within: body div div0 div1 div2 div3 div4 div5 div6 div7 epilogue group lg performance prologue
Members of: byline closer epigraph salute signed trailer

```
<!ENTITY % m.divbot '
  %x.divbot trailer | signed | salute | epigraph | closer |
  byline'                                                    >
```

Discussed in: 7.2 ('Elements Common to All Divisions') on p. 190

divn

Class Name: divn
Description: structural elements which behave in the same way as divisions.
Attributes:

type specifies a name conventionally used for this level of subdivision, e.g. "act", "volume", "book", "section", "canto", etc.
Data type: CDATA
Value: any string of characters
Default: #CURRENT

org specifies how the content of the division is organized.
Data type: (COMPOSITE | UNIFORM)
Sample values include:

composite composite content: i.e. no claim is made about the sequence in which

the immediate contents of this division are to be processed, or their interrelationships.

uniform uniform content: i.e. the immediate contents of this element are regarded as forming a logical unit, to be processed in sequence.

Default: UNIFORM

sample indicates whether this division is a sample of the original source and if so, from which part.

Data type: (INITIAL | MEDIAL | FINAL | UNKNOWN | COMPLETE)

Sample values include:

initial division lacks material present at end in source.

medial division lacks material at start and end.

final division lacks material at start.

unknown position of sampled material within original unknown.

complete division is not a sample.

Default: COMPLETE

part specifies whether or not the division is fragmented by some other structural element, for example a speech which is divided between two or more verse stanzas.

Data type: (Y | N | I | M | F)

Sample values include:

Y the division is incomplete in some respect

N either the division is complete, or no claim is made as to its completeness.

I the initial part of an incomplete division

M a medial part of an incomplete division

F the final part of an incomplete division

Default: N

The values I, M, or F should be used only where it is clear how the division is to be reconstituted.

Part: base tag set for common core features
Member of classes:
DTD file: teiclas2.ent
Occurs within: [none]
Members of: div div0 div1 div2 div3 div4 div5 div6 div7 lg lg1 lg2 lg3 lg4 lg5

```
<!ENTITY % a.divn '
          type              CDATA               #CURRENT
          org               (composite | uniform)
                                                uniform
          sample            (initial | medial | final |
                            unknown | complete) complete
          part              (Y | N | I | M | F) N'            >
```

Discussed in: 7 ('Default Text Structure') on p. 183

Class Name: divtop (i.e. top-of-div elements) divtop

Description: groups elements which can occur at the start of a text block: head, epigraph, byline, etc.
Attributes: [None: global and inherited attributes only.]
Part: base tag set for common core features
Member of classes:
DTD file: teistr2
Occurs within: back body castList div div0 div1 div2 div3 div4 div5 div6 div7 epilogue front group lg performance prologue
Members of: argument byline docAuthor docDate epigraph head opener salute signed

```
<!ENTITY % m.divtop '
  %x.divtop signed | salute | opener | head | epigraph | docDate
  | docAuthor | byline | argument'                              >
```

Discussed in: 7.2 ('Elements Common to All Divisions') on p. 190

dramafront

Class Name: dramafront

Description: elements which appear at the level of divisions within front or back matter of performance texts only.
Attributes: [None: global and inherited attributes only.]
Remarks:
Part: base tag set for performance texts
Member of classes: front
DTD file: teidram2.ent
Occurs within: back front
Members of: epilogue performance prologue set

```
<!ENTITY % m.dramafront '
  %x.dramafront set | prologue | performance | epilogue'        >
```

Discussed in: 10.1 ('Front and Back Matter ') on p. 228

edit

Class Name: edit

Description: phrase-level elements for simple editorial correction and transcription.
Attributes: [None: global and inherited attributes only.]
Remarks:
Part: base tag set for common core features
Member of classes: phrase
DTD file: teiclas2.ent
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
Members of: add corr del gap orig reg sic

```
<!ENTITY % m.edit '
  %x.edit sic | reg | orig | gap | del | corr | add'           >
```

Discussed in: 6.5 ('Simple Editorial Changes') on p. 140

edit

Class Name: edit

Description: phrase-level elements for simple editorial correction and transcription.
Attributes:

resp (i.e. responsible) signifies the editor or transcriber responsible for the salient information conveyed by a particular tag: the hand of an addition or deletion, the expansion of an abbreviation, the correction of an apparent error, the regularization of a non-standard form, the transcription of unclear material, or the decision not to transcribe some portion of the text.
Data type: IDREF
Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).
Default: %INHERITED
As noted, the precise type of responsibility exercised by the individual named in the attribute varies with the particular element type. Responsibility for other aspects of the

markup may be recorded using the methods described in chapter 17 ('Certainty and Responsibility') on p. 435.

cert (i.e. certainty) signifies the degree of certainty ascribed to some specific aspect of the markup: the identification of the hand of an addition or deletion, the correctness of the expansion of an abbreviation, the correction of an error, or the regularization of a non-standard form; or the correctness of the transcription of unclear material.
Data type: CDATA
Default: #IMPLIED

Remarks:
This version of this class is used only when the additional tag set for transcription of primary sources is used.

Part: base tag set for common core features
Member of classes: phrase
DTD file: `teitran2.ent`
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
Members of:

```
<!ENTITY % m.edit '
  %x.edit sic | reg | orig | gap | del | corr | add'           >
<!ENTITY % a.edit '
          resp              IDREF              %INHERITED
          cert              CDATA              #IMPLIED'      >
```

Discussed in: 18.1.1 ('Use of Core Tags for Transcriptional Work') on p. 445

Class Name: `enjamb` (i.e. enjambement) enjamb

Description: elements bearing the `enjamb` attribute.
Attributes:

enjamb (i.e. enjambement) indicates that the end of a verse line is marked by enjambement.
Data type: CDATA
Sample values include:

no the line is end-stopped
yes the line in question runs on into the next
weak the line is weakly enjambed
strong the line is strongly enjambed

Default: #IMPLIED
The usual practice will be to give the value "y" to this attribute when enjambement is being marked, or the values "weak" and "strong" if degrees of enjambement are of interest; if no value is given, however, the attribute does not default to a value of "no"; this allows the attribute to be omitted entirely when enjambement is not of particular interest.

Part: base tag set for verse texts
Member of classes:
DTD file: `teivers2.ent`
Occurs within: [none]
Members of: l

```
<!ENTITY % a.enjamb '
          enjamb              CDATA               #IMPLIED'      >
```

Discussed in: 9.3 ('Components of the Verse Line') on p. 218

entries

Class Name: entries (i.e. dictionary entries)

Description: groups the different styles of dictionary entries.

Attributes:

type indicates type of entry, in dictionaries with multiple types.

Data type: CDATA

Sample values include:

main a main entry (default).

hom a homograph with a separate entry.

xref a reduced entry whose only function is to point to another main entry (e.g. for forms of an irregular verb or for variant spellings: 'was' pointing to 'be', or 'esthete' to 'aesthete').

affix an entry for a prefix, infix, or suffix.

abbr an entry for an abbreviation.

supplemental a supplemental entry (for use in dictionaries which issue supplements to their main work in which they include updated information about entries).

foreign an entry for a "foreign" word in a monolingual dictionary.

Default: "MAIN"

key (i.e. sort key) contains a (sortable) character sequence reflecting the entry's alphabetical position in the printed dictionary.

Data type: CDATA

Value: any sequence of characters which, when sorted with the other values, will produced the desired order; specifics of key construction are application-dependent.

Default: #IMPLIED

Dictionary order often differs from the collation sequence of machine-readable character sets; in English-language dictionaries, an entry for '4-H' will often appear alphabetized under "fourh", and 'McCoy' may be alphabetized under "maccoy", while 'A1', 'A4', and 'A5' may all appear in numeric order "alphabetized" between "a-" and "AA". The sort key is required if the orthography of the dictionary entry does not suffice to determine its location.

Remarks:

The global n attribute should be used to encode the homograph numbers attached to entries for homographs.

Part: base tag set for printed dictionaries

Member of classes:

DTD file: teidict2.ent

Occurs within: [none]

Members of: entry entryFree superentry

```
<!ENTITY % a.entries '
          type                CDATA               "main"
          key                 CDATA               #IMPLIED'      >
```

Discussed in: 12.1 ('Dictionary Body and Overall Structure') on p. 270; 12.2 ('The Structure of Dictionary Entries') on p. 274

featureVal

Class Name: featureVal (i.e. feature values)

Description: elements which express feature values in feature structures.

Attributes: [None: global and inherited attributes only.]

Part: additional tag set for feature structures

Member of classes:

DTD file: teifsd2

Occurs within: if vDefault vRange

Members of: null *singleVal* (dft msr nbr rate str sym uncertain *boolean* (any none) *binary* (minus plus)) *complexVal* (alt fs vAlt)

```
<!ENTITY % m.featureVal '
  %x.featureVal %m.singleVal | %m.complexVal | null'          >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

Class Name: formInfo (i.e. form information) formInfo

Description: elements allowed within a <form> element in a dictionary.
Attributes: [None: global and inherited attributes only.]
Part: base tag set for printed dictionaries
Member of classes:
DTD file: teidict2.ent
Occurs within: form
Members of: form hyph lbl orth pron syll usg

```
<!ENTITY % m.formInfo '
  %x.formInfo usg | syll | pron | orth | lbl | hyph | form'     >
```

Discussed in: 12.3.1 ('Information on Written and Spoken Forms') on p. 279

Class Name: formPointers (i.e. form pointers) formPointers

Description: groups elements in the dictionary base which point at orthographic or pronunciation forms of the headword.
Attributes:

target gives the SGML identifier of the orthographic form referred to.
Data type: IDREF
Value: an ID value on some <orth> or <form> element.
Default: #IMPLIED

Remarks:
(optional)

Part: base tag set for printed dictionaries
Member of classes: phrase
DTD file: teidict2.ent
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref [May appear anywhere within: eg eLeaf]
Members of: oRef oVar pRef pVar

```
<!ENTITY % m.formPointers '
  %x.formPointers pVar | pRef | oVar | oRef'                  >
<!ENTITY % a.formPointers '
          target             IDREF               #IMPLIED'    >
```

Discussed in: 12 ('Print Dictionaries') on p. 269

Class Name: fragmentary fragmentary

Description: elements which mark the beginning or ending of a fragmentary manuscript or other witness.
Attributes:

wit (i.e. witnesses) contains a list of one or more sigla indicating the witnesses which begin or end at this point.
Data type: CDATA
Value: A space-delimited series of sigla; each sigil should correspond to a witness or witness group and occur as the value of the sigil attribute on a <witness> element elsewhere in the document.
Default: #IMPLIED
Example:

```
<rdg wit='M N'>Exper<witStart wit='M'>ience</rdg>
```

Remarks:
These elements may appear only within text-critical apparatus.

Part: additional tag set for text criticism
Member of classes:
DTD file: teitc2.ent
Occurs within: [none] [May appear anywhere within: lem rdg]
Members of: lacunaEnd lacunaStart witEnd witStart

```
<!ENTITY % m.fragmentary '
  %x.fragmentary witStart | witEnd | lacunaStart | lacunaEnd'   >
<!ENTITY % a.fragmentary '
          wit                CDATA               #IMPLIED'      >
```

Discussed in: 19.1.5 ('Fragmentary Witnesses') on p. 479

front

Class Name: front

Description: elements which appear at the level of divisions within front or back matter.
Attributes: [None: global and inherited attributes only.]
Remarks:

Part: base tag set for common core features
Member of classes:
DTD file: teiclas2.ent
Occurs within: back front
Members of: titlePage *dramafront* (epilogue performance prologue set)

```
<!ENTITY % m.front '
  %x.front %m.dramafront | titlePage'                          >
```

Discussed in: 10.1 ('Front and Back Matter ') on p. 228

global

Class Name: global

Description: defines attributes common to all elements in the TEI encoding scheme.
Attributes:

id (i.e. identifier) provides a unique identifier for the element bearing the ID value.
Data type: ID
Value: any valid SGML name (in the reference concrete syntax, a name must begin with a letter and contain only letters, numeric digits, hyphen, and full stop).
Default: #IMPLIED
Example:

```
<p id=names>Paragraph with the ID <mentioned>names</>....
<p id=dates>Paragraph with the ID <mentioned>dates</>....
```

The id attribute may be used to specify a canonical reference for an element; see section 6.9 ('Reference Systems') on p. 155.

n (i.e. number, name, etc.) gives a number (or other label) for an element, which is not necessarily unique within the document.
Data type: CDATA
Value: any string of characters; often, but not necessarily, numeric.
Default: #IMPLIED

The n attribute may be used to specify the numbering of chapters, sections, list items, etc.; it may also be used in the specification of a standard reference system for the text.

lang (i.e. language) indicates the language of the element content, usually using a two- or three-letter code from ISO 639.
Data type: IDREF
Value: The value must be the identifier specified for a writing system declaration declared in the TEI header, as described in section 5 ('The TEI Header') on p. 77.
Default: %INHERITED
Example:

```
<p lang=EN>The only surviving work by <rs>Ari</></rs>
(died 1148) is the ten-page <title
lang=IS>&Iacute;slendingab&oacute;k</> (<title
lang=LA>Libellus Islandorum</>), written in the early
twelfth century.
```

If no value is specified for lang, the lang value for the immediately enclosing element is inherited; for this reason, a value should always be specified on the outermost element (<tei.2>).

rend (i.e. rendition or presentation) indicates how the element in question was rendered or presented in the source text.
Data type: CDATA
Value: any string of characters; if the typographic rendition of a text is to be systematically recorded, a systematic set of values for the rend attribute should be defined.
Default: #IMPLIED
These Guidelines make no binding recommendations for the values of the rend attribute; the characteristics of visual presentation vary too much from text to text and the decision to record or ignore individual characteristics varies too much from project to project. Some potentially useful conventions are noted from time to time at appropriate points in the Guidelines.

Remarks:
The global attributes described here are made part of the attribute definition list declaration of each element by including the string "%a.global" in each such declaration. Some global attributes are made available when certain base or additional tag sets are selected; these are incorporated into the global attributes by references to the appropriate parameter entities. When the tag sets in question have not been selected, the parameter entities in question expand to the empty string.

Part: base tag set for common core features
Member of classes: analysis, linking, terminology
DTD file: teiclas2.ent
Occurs within: [none]
Members of:

```
<!ENTITY % a.global '        %a.analysis
                             %a.linking
                             %a.terminology
          id                 ID                  #IMPLIED
          n                  CDATA               #IMPLIED
          lang               IDREF               %INHERITED
          rend               CDATA               #IMPLIED'      >
```

Discussed in: 3.5 ('Global Attributes') on p. 42

Class Name: global

global

Description: defines global attributes for the writing system declaration.
Attributes:

id gives a unique identifier for the element.
Data type: ID
Value: any valid SGML name unique within the document.

Default: #IMPLIED

lang (i.e. language) gives the language in which the content of the element is written.
Data type: CDATA
Value: Should be a language code from ISO 639.
Default: %INHERITED
This attribute functions like the global attribute of the same name in the main TEI DTD; for technical reasons it is declared differently.

Remarks:

Part: auxiliary tag set for writing system declarations
Member of classes:
DTD file: teiwsd2
Occurs within: [none]
Members of:

```
<!ENTITY % a.global '
          id                  ID                  #IMPLIED
          lang                CDATA               %INHERITED'    >
```

Discussed in: 25.1 ('Overall Structure of Writing System Declaration') on p. 571

globincl

Class Name: globincl (i.e. global inclusions)
Description: empty elements which may appear at any point within a TEI text.
Attributes: [None: global and inherited attributes only.]

Remarks:

Part: base tag set for common core features
Member of classes:
DTD file: teiclas2.ent
Occurs within: [none] [May appear anywhere within: text]
Members of: *metadata* (certainty interp interpGrp respons span spanGrp)

```
<!ENTITY % m.globincl '
  %x.globincl %m.metadata |'                                    >
```

Discussed in: 3.7 ('Element Classes') on p. 51

gramInfo

Class Name: gramInfo (i.e. grammatical information)
Description: elements allowed within a <gramGrp> element in a dictionary.
Attributes: [None: global and inherited attributes only.]
Part: base tag set for printed dictionaries
Member of classes:
DTD file: teidict2.ent
Occurs within: gramGrp
Members of: colloc gramGrp lbl pos subc usg

```
<!ENTITY % m.gramInfo '
  %x.gramInfo usg | subc | pos | lbl | gramGrp | colloc'        >
```

Discussed in: 12.3.2 ('Grammatical Information') on p. 284

hqinter

Class Name: hqinter
Description: intermediate-level elements related to highlighting.
Attributes: [None: global and inherited attributes only.]

Remarks:

Part: base tag set for common core features
Member of classes: common, inter
DTD file: teiclas2.ent
Occurs within: add admin argument body camera caption case castList cell colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood

note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref

Members of: cit q quote

```
<!ENTITY % m.hqinter '
  %x.hqinter quote | q | cit'                                    >
```

Discussed in: 6.3 ('Highlighting and Quotation') on p. 123

Class Name: hqphrase hqphrase

Description: phrase-level elements related to highlighting.
Attributes: [None: global and inherited attributes only.]

Remarks:

Part: base tag set for common core features
Member of classes: phrase
DTD file: teiclas2.ent
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

Members of: distinct emph foreign gloss hi mentioned soCalled term title

```
<!ENTITY % m.hqphrase '
  %x.hqphrase title | term | soCalled | mentioned | hi | gloss |
  foreign | emph | distinct'                                      >
```

Discussed in: 6.3 ('Highlighting and Quotation') on p. 123

Class Name: inter inter

Description: elements of the intermediate (inter-level) class, which can occur both within paragraphs and between paragraphs.
Attributes: [None: global and inherited attributes only.]

Remarks:

This element class contains a subset of those elements which can appear in the unstructured "soup" with which paragraph and other elements at the lowest level of crystal structures are filled: specifically all the elements which can also occur as structural elements in their own right. In prose, this means the elements in this class can appear both within and between paragraphs. This class is thus distinct from the purely phrase-level elements which can appear only within soup, and not on their own; the latter class, in keeping with this metaphor, is called "broth"; it is represented by the class *phrase*. Cf. also the class *chunks*.

Part: base tag set for common core features
Member of classes:
DTD file: teiclas2.ent
Occurs within: add admin camera caption case cell colloc corr country damage def desc descrip docEdition emph equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting mood note number orth otherForm p per pos pron q quote rdg ref region rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref [May appear anywhere within: entryFree]

Members of: castList figure stage table text *stageDirection* (camera caption move sound tech view) *notes* (note) *lists* (label list listBibl) *hqinter* (cit q quote) *bibl* (bibl biblFull biblStruct)

```
<!ENTITY % m.inter '
  %x.inter %m.stageDirection | %m.notes | %m.lists | %m.hqinter
  | %m.bibl | text | table | stage | figure | castList'        >
```

Discussed in: 3.7 ('Element Classes') on p. 51

interpret

Class Name: interpret

Description: defines attributes common to all interpretative elements.

Attributes:

resp indicates who is responsible for the interpretation.
Data type: CDATA
Value: Any string of characters, such as the initials of the encoder.
Default: %INHERITED

type indicates what kind of phenomenon is being noted in the passage.
Data type: CDATA
Sample values include:
image identifies an image in the passage.
character identifies a character associated with the passage.
theme identifies a theme in the passage.
allusion identifies an allusion to another text.
(discourse type) specifies that the passage is of a particular discourse type.
Default: %INHERITED

inst points to instances of the analysis or interpretation represented by the current element.
Data type: IDREFS
Value: One or more valid SGML identifiers.
Default: #IMPLIED
The current element should be an analytic one. The element pointed at should be a textual one.

Part: additional tag set for analysis and interpretation

Member of classes:

DTD file: teiclas2.ent

Occurs within: [none]

Members of: interp interpGrp span spanGrp

```
<!ENTITY % a.interpret '
          resp              CDATA             %INHERITED
          type              CDATA             %INHERITED
          inst              IDREFS            #IMPLIED'      >
```

Discussed in: 15.2 ('Global Attributes for Simple Analyses') on p. 387

linking

Class Name: linking

Description: default declaration for class *linking*: when the additional tag set for linking is not selected, no attributes are defined for this class.

Attributes: [None: global and inherited attributes only.]

Part: additional tag set for analysis and interpretation

Member of classes:

DTD file: teiclas2.ent

Occurs within: [none]

Members of:

```
<!ENTITY % a.linking ''                                          >
```

Discussed in: 3.5 ('Global Attributes') on p. 42; 14 ('Linking, Segmentation, and Alignment') on p. 331

linking

Class Name: linking

Description: defines additional attributes for hypertext and other linking, which are enabled for all elements when the additional tag set for linking is selected.

Attributes:

corresp (i.e. correspondents) points to elements that correspond to the current element in some way.
Data type: IDREFS
Value: one or more valid SGML identifiers for elements in the current document, separated by white space.
Default: #IMPLIED

synch (i.e. synchronous) points to elements that are synchronous with the current element.
Data type: IDREFS
Value: one or more valid SGML identifiers for elements in the current document, separated by white space.
Default: #IMPLIED

sameAs points to an element that is the same as the current element.
Data type: IDREF
Value: a valid SGML identifier.
Default: #IMPLIED

copyOf points to an element of which the current element is a copy.
Data type: IDREF
Value: a valid SGML identifier for an element in the current document.
Default: #IMPLIED
Any content of the current element should be ignored. Its true content is that of the element being pointed at.

next points to the next element of a virtual aggregate of which the current element is part.
Data type: IDREF
Value: a valid SGML identifier.
Default: #IMPLIED

prev points to the previous element of a virtual aggregate of which the current element is part.
Data type: IDREF
Value: a valid SGML identifier.
Default: #IMPLIED

exclude points to elements that are in exclusive alternation with the current element.
Data type: IDREFS
Value: a list of valid SGML identifiers.
Default: #IMPLIED

select selects one or more alternants; if one alternant is selected, the ambiguity or uncertainty is marked as resolved. If more than one alternant is selected, the degree of ambiguity or uncertainty is marked as reduced by the number of alternants not selected.
Data type: IDREFS
Value: a list of valid SGML identifiers.
Default: #IMPLIED
This attribute should be placed on an element which is superordinate to all of the alternants from which the selection is being made.

Part: additional tag set for analysis and interpretation
Member of classes:
DTD file: `teilink2.ent`
Occurs within: [none]
Members of: *global* ()

```
<!ENTITY % a.linking '
          corresp           IDREFS            #IMPLIED
          synch             IDREFS            #IMPLIED
          sameAs            IDREF             #IMPLIED
          copyOf            IDREF             #IMPLIED
          next              IDREF             #IMPLIED
          prev              IDREF             #IMPLIED
          exclude           IDREFS            #IMPLIED
          select            IDREFS            #IMPLIED'      >
```

Discussed in: 14 ('Linking, Segmentation, and Alignment') on p. 331

lists

Class Name: lists

Description: list-like elements.
Attributes: [None: global and inherited attributes only.]

Remarks:

Part: base tag set for common core features
Member of classes: common, inter
DTD file: `teiclas2.ent`
Occurs within: add admin argument body camera caption case castList cell colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref
Members of: label list listBibl

```
<!ENTITY % m.lists '
  %x.lists listBibl | list | label'                    >
```

Discussed in: 6.7 ('Lists') on p. 149

loc

Class Name: loc

Description: elements used for purposes of location and reference
Attributes: [None: global and inherited attributes only.]

Remarks:

Part: base tag set for common core features
Member of classes: phrase
DTD file: `teiclas2.ent`
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cit cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
Members of: link ptr ref xptr xref

```
<!ENTITY % m.loc '
  %x.loc xref | xptr | ref | ptr | link'                >
```

Discussed in: 6.6 ('Simple Links and Cross References') on p. 147

metadata

Class Name: metadata

Description: empty elements which describe the status of other elements, for example by holding groups of links or of abstract interpretations, or by providing indications of certainty etc., and which may appear at any point in a document.
Attributes: [None: global and inherited attributes only.]

Remarks:
Encoders may find it convenient to localize all metadata elements, for example to contain them within the same divison as the elements that they relate to; or to locate them all to a division of their own. They may however appear at any point in a TEI text.

Part: base tag set for common core features

Member of classes: globincl
DTD file: teiclas2.ent
Occurs within: [none]
Members of: certainty interp interpGrp respons span spanGrp

```
<!ENTITY % m.metadata '
  %x.metadata spanGrp | span | respons | interpGrp | interp |
  certainty'                                                    >
```

Discussed in: 3.7 ('Element Classes') on p. 51

Class Name: metrical

metrical

Description: elements carrying metrical information.
Attributes:

met contains a user-specified encoding for the conventional metrical structure of the element.
Data type: CDATA
Value: May contain either a standard term for the kind of metrical unit (e.g. hexameter) or an encoded representation for the metrical pattern (e.g. +-+-+-+-+-). In either case, the notation used should be documented by a <metNotation> element within the <encodingDesc> of the associated header.
Default: %INHERITED
Where this attribute is not specified, the metrical pattern for the element concerned is understood to be inherited from its parent.

real contains a user-specified encoding for the actual realization of the conventional metrical structure applicable to the element.
Data type: CDATA
Value: May contain either a standard term for the kind of metrical unit (e.g. "hexameter") or an encoded representation for the metrical pattern (e.g. "+–+-+-+-+-"). In either case, the notation used should be documented by a <metNotation> element within the <encodingDesc> of the associated header.
Default: #IMPLIED
Where this attribute is not specified, the metrical realization for the element concerned is understood to be identical to that specified or implied for the met attribute.

rhyme specifies the rhyme scheme applicable to a group of verse lines.
Data type: CDATA
Value: By default, the rhyme scheme is expressed as a string of alphabetic characters each corresponding with a rhyming line. Any non-rhyming lines should be represented by a hyphen or an X. Alternative notations may be defined as for met by use of the <metNotation> element in the TEI header.
Default: #IMPLIED
Example:

```
<lg rhyme='ABABABCC'> ... </lg>
```

When the default notation is used, it does not make sense to specify this attribute on any unit smaller than a line. Nor does the default notation provide any way to record internal rhyme, or to specify non-conventional rhyming practice. These extensions would require user-defined alternative notations.

Remarks:

Part: base tag set for verse texts
Member of classes:
DTD file: teivers2.ent
Occurs within: [none]
Members of: l lg lg1 lg2 lg3 lg4 lg5

```
<!ENTITY % a.metrical '      %a.global
          met                CDATA                %INHERITED
```

```
            real              CDATA             #IMPLIED
            rhyme             CDATA             #IMPLIED'      >
```

Discussed in: 9.4 ('Rhyme and Metrical Analysis') on p. 221

morphInfo

Class Name: morphInfo (i.e. morphological elements)

Description: groups elements for providing morphological information.
Attributes: [None: global and inherited attributes only.]

Remarks:

Part: base tag set for printed dictionaries
Member of classes:
DTD file: teidict2.ent
Occurs within: etym
Members of: case gen gram itype mood number per tns

```
<!ENTITY % m.morphInfo '
  %x.morphInfo tns | per | number | mood | itype | gram | gen |
  case'                                                          >
```

Discussed in: 12.3 ('Top-level Constituents of Entries') on p. 279

names

Class Name: names

Description: elements with proper nouns as content.
Attributes:

key provides an alternative identifier for the object being named, such as a database record key.
Data type: CDATA
Value: any string
Default: #IMPLIED
Example:

```
<rs type='place'>Montaillou</rs>
is not a large parish.
At the time of the events which led to
<rs type='person' key=BXII></rs>
Fournier's</name> investigations,
the local population consisted of between 200 and
250 inhabitants.
```

The value may be a unique identifier from a database, or simply a more explicit name for the referent. Its purpose is only to record an identification; if the analysis leading to the identification is to be recorded as well, the analytic tags described in chapter 16 ('Feature Structures') on p. 397 should be used in addition or instead.

reg (i.e. regularization) gives a normalized or regularized form of the name used.
Data type: CDATA
Value: Any string of characters.
Default: #IMPLIED
Example:

```
At the time of the events which led to
<rs type=person reg='Benedict XII, Pope of Avignon</rs>
(Jacques Fournier)'>Fournier's</name> investigations,
the local population consisted of between 200 and
250 inhabitants.
```

In providing a "regularized" form, no claim is made that the form in the source text is incorrect; the regularized form is simply that chosen as the main form for purposes of unifying variant forms under a single heading.

Remarks:

Part: base tag set for common core features
Member of classes:

DTD file: teind2.ent
Occurs within: [none]
Members of: measure name persName placeName pubPlace rs *placePart* (bloc country distance geog geogName offset placeName region settlement) *personPart* (addName forename genName nameLink roleName surname)

```
<!ENTITY % a.names '          %a.global
          key                CDATA                #IMPLIED
          reg                CDATA                #IMPLIED'      >
```

Discussed in: 6.4.1 ('Referring Strings') on p. 132

Class Name: notes notes

Description: note-like elements.
Attributes: [None: global and inherited attributes only.]

Remarks:

Part: base tag set for common core features
Member of classes: common, inter
DTD file: teiclas2.ent
Occurs within: add admin argument body camera caption case castList cell colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref
Members of: note

```
<!ENTITY % m.notes '
  %x.notes note'                                              >
```

Discussed in: 6.8 ('Notes, Annotation, and Indexing') on p. 152

Class Name: personPart (i.e. components of personal names) personPart

Description: groups elements which form part of a personal name.
Attributes:

type provides more culture- linguistic- or application- specific information used to categorize this name component.
Data type: CDATA
Value: one of a set of codes defined for the application.
Default: #IMPLIED

full indicates whether the name component is given in full, as an abbreviation or simply as an initial.
Data type: (YES | ABB | INIT)
Sample values include:

yes the name component is spelled out in full.
abb the name component is given in an abbreviated form.
init the name component is indicated only by one initial.

Default: YES

sort specifies the sort order of the name component in relation to others within the personal name.
Data type: NUMBER
Value: A positive number indicating the sort order.
Default: #IMPLIED

Part: additional tag set for names and dates
Member of classes: names
DTD file: teind2.ent
Occurs within: persName
Members of: addName forename genName nameLink roleName surname

```
<!ENTITY % m.personPart '
  %x.personPart surname | roleName | nameLink | genName |
  forename | addName'                                          >
<!ENTITY % a.personPart '    %a.global
                             %a.names
          type               CDATA                #IMPLIED
          full               (yes | abb | init)   yes
          sort               NUMBER               #IMPLIED'     >
```

Discussed in: 20.1 ('Personal Names') on p. 488

phrase

Class Name: phrase

Description: includes elements which can occur at the level of individual words or phrases.
Attributes: [None: global and inherited attributes only.]

Remarks:
This class of elements can occur only within larger elements of the class *inter* or *chunk*. In prose, this means these elements can occur within paragraphs, list items, lines of verse, etc.

Part: base tag set for common core features
Member of classes:
DTD file: teiclas2.ent
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref [May appear anywhere within: entryFree]
Members of: formula handShift *sgmlKeywords* (att gi tag val) *seg* (anchor c cl m phr s seg w) *phrase.verse* (caesura) *loc* (link ptr ref xptr xref) *hqphrase* (distinct emph foreign gloss hi mentioned soCalled term title) *formPointers* (oRef oVar pRef pVar) *edit* () *edit* (add corr del gap orig reg sic) *data* (abbr address date dateRange dateStruct expan lang measure name num rs time timeRange timeStruct)

```
<!ENTITY % m.phrase '
  %x.phrase %m.sgmlKeywords | %m.seg | %m.phrase.verse | %m.loc
  | %m.hqphrase | %m.formPointers | %m.edit | %m.edit | %m.data
  | handShift | formula'                                       >
```

Discussed in: 3.7 ('Element Classes') on p. 51

phrase.verse

Class Name: phrase.verse

Description: phrase-level elements which may appear within verse only.
Attributes: [None: global and inherited attributes only.]

Remarks:

Part: additional tag set for verse texts
Member of classes: phrase
DTD file: teivers2.ent
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName

orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

Members of: caesura

```
<!ENTITY % m.phrase.verse '
  %x.phrase.verse caesura'                                    >
```

Discussed in: 9.3 ('Components of the Verse Line') on p. 218

Class Name: placePart (i.e. place name components) placePart

Description: groups elements which form part of a place name.

Attributes:

type provides more culture- linguistic- or application- specific information used to categorize this name component.
Data type: CDATA
Value: one of a set of codes defined for the application.
Default: #IMPLIED

full indicates whether the place name component is given in full, as an abbreviation or simply as an initial
Data type: (YES | ABB | INIT)
Sample values include:

yes the name component is spelled out in full.

abb the name component is given in an abbreviated form.

init the name component is indicated only by one initial.

Default: YES

Part: additional tag set for names and dates

Member of classes: names

DTD file: teind2.ent

Occurs within: placeName

Members of: bloc country distance geog geogName offset placeName region settlement

```
<!ENTITY % m.placePart '
  %x.placePart settlement | region | placeName | offset |
  geogName | geog | distance | country | bloc'                  >
<!ENTITY % a.placePart '     %a.global
                             %a.names
          type               CDATA               #IMPLIED
          full               (yes | abb | init)  yes'           >
```

Discussed in: 20.2 ('Place Names') on p. 493

Class Name: pointer pointer

Description: elements which point to other elements, using IDREFs.

Attributes:

type categorizes the pointer in some respect, using any convenient set of categories.
Data type: CDATA
Value: The type should indicate the intended function of the pointer, or the rhetorical relationship between its source and target.
Default: #IMPLIED

resp specifies the creator of the pointer.
Data type: CDATA
Value: any string of characters, usually the initials or name of the creator.
Default: #IMPLIED

crdate specifies when the pointer was created.
Data type: CDATA

Value: any string representing a date.
Default: #IMPLIED

targType specifies the kinds of elements to which this pointer may point.
Data type: NAMES
Value: A list of names of SGML elements defined in the DTD of the current document.
Default: #IMPLIED
If this attribute is supplied, every element specified as a target must be of one or other of the types specified. An application may choose whether or not to report failures to satisfy this constraint as errors, but may not access an element of the right identifier but the wrong type.

targOrder where more than one identifier is supplied as the value of the target attribute, this attribute specifies whether the order in which they are supplied is significant.
Data type: (Y | N | U)
Sample values include:

Y Yes: the order in which IDREFs are specified as the value of a target attribute should be followed when combining the targeted elements.

N No: the order in which IDREFs are specified as the value of a target attribute has no significance when combining the targeted elements.

U Unspecified: the order in which IDREFs are specified as the value of a target attribute may or may not be significant.

Default: U

evaluate specifies the intended meaning when the target of a pointer is itself a pointer.
Data type: (ALL | ONE | NONE)
Sample values include:

all if the element pointed to is itself a pointer, then the target of that pointer will be taken, and so on, until an element is found which is not a pointer.

one if the element pointed to is itself a pointer, then its target (whether a pointer or not) is taken as the target of this pointer.

none no further evaluation of targets is carried out beyond that needed to find the element specified in the pointer's target.

Default: #IMPLIED
If no value is given, the application program is responsible for deciding (possibly on the basis of user input) how far to trace a chain of pointers.

Remarks:

Part: base tag set for common core features
Member of classes:
DTD file: teilink2.ent
Occurs within: [none]
Members of: alt join link ptr ref *xPointer* (xptr xref) *pointerGroup* (altGrp joinGrp linkGrp)

```
<!ENTITY % a.pointer '       %a.global
          type               CDATA               #IMPLIED
          resp               CDATA               #IMPLIED
          crdate             CDATA               #IMPLIED
          targType           NAMES               #IMPLIED
          targOrder          (Y | N | U)         U
          evaluate           (all | one | none)  #IMPLIED'      >
```

Discussed in: 6.6 ('Simple Links and Cross References') on p. 147

pointerGroup

Class Name: pointerGroup
Description: elements which enclose groups of pointer elements.
Attributes:

domains optionally specifies the identifiers of the elements within which all elements indicated by the contents of this element lie.
Data type: IDREFS

Value: a list of at least two valid SGML identifiers.
Default: #IMPLIED
Example:

```
<linkGrp type='imitation'
         domains='dunciad dunnotes'
         targType='note l'
         targOrder=Y>
```

If this attribute is supplied every element specified as a target must be contained within the element or elements specified by it. An application may choose whether or not to report failures to satisfy this constraint as errors, but may not access an element of the right identifier but in the wrong context. If this attribute is not supplied, then target elements may appear anywhere within the current document.

targFunc describes the function of each of the values of the targets attribute of the enclosed <link>, <join> or <alt> tags.
Data type: NMTOKENS
Value: a list of at least two valid SGML name tokens.
Default: #IMPLIED
Example:

```
<linkGrp resp=NPR
         type='imitation'
         domains='dunciad dunnotes dunnotes'
         targType='note ref l'
         targFunc='source reference.to.goal goal'
         targOrder=Y>
```

The number of separate values should match the number of values in the targets attribute in the enclosed <link>, <join> or <alt> tags (unless the extendTarg attribute allows the targets attributes of those tags to have more values). It should also match the number of values in the targType and domains attributes of the current tag, if those have been specified.

Part: additional tag set for
Member of classes: pointer
DTD file: teilink2.ent
Occurs within: [none]
Members of: altGrp joinGrp linkGrp

```
<!ENTITY % a.pointerGroup '  %a.global
                             %a.pointer
           domains           IDREFS              #IMPLIED
           targFunc          NMTOKENS            #IMPLIED'      >
```

Discussed in: 14 ('Linking, Segmentation, and Alignment') on p. 331

Class Name: readings

readings

Description: elements representing variant readings in text critical work.
Attributes:

wit (i.e. witnesses) contains a list of one or more sigla indicating the witnesses which attest to a given reading.
Data type: CDATA
Value: A space-delimited series of sigla; each sigil should correspond to a witness or witness group and occur as the value of the sigil attribute on a <witness> element elsewhere in the document.
Default: #IMPLIED
Example:

```
<rdg wit='El Hg'>Experience</rdg>
```

If the apparatus contains readings only for a single witness, this attribute may be consistently omitted.

type classifies the reading according to some useful typology.
Data type: CDATA
Sample values include:
substantive the reading offers a substantive variant.
orthographic the reading differs only orthographically, not in substance, from other readings.
Default: #IMPLIED

cause classifies the reading as original or non-original, according to some typology of possible origins.
Data type: CDATA
Value: any word or phrase describing the cause: e.g. "homeoteleuton", "homeoarchy", "paleographic confusion", "haplography", "dittography", "false emendation".
Default: #IMPLIED

varSeq (i.e. variant sequence) provides a number indicating the position of this reading in a sequence, when there is reason to presume a sequence to the variants on any one lemma.
Data type: NUMBER
Value: a positive integer
Default: #IMPLIED
Different variant sequences could be coded with distinct number trails: 1-2-3 for one sequence, 5-6-7 for another. More complex variant sequences, with (for example) multiple branchings from single readings, may be expressed through the <join> element.

resp (i.e. responsibility) identifies the editor responsible for asserting a particular reading in the witness.
Data type: CDATA
Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).
Default: %INHERITED
This attribute is only available within an apparatus gathering variant readings in the transcription of an individual witness. It may not occur in an apparatus gathering readings from different witnesses.

hand signifies the hand responsible for a particular reading in the witness.
Data type: IDREF
Value: must be one of the hand identifiers declared in the document header (see section 18.2.1 ('Document Hands') on p. 456).
Default: %INHERITED
This attribute is only available within an apparatus gathering variant readings in the transcription of an individual witness. It may not occur in an apparatus gathering readings from different witnesses.

wit (i.e. witness) contains a list of one or more sigla of witnesses attesting a given reading.
Data type: CDATA
Value: the list of sigla.
Default: #IMPLIED
This attribute may occur both within an apparatus gathering variant readings in the transcription of an individual witness and within an apparatus gathering readings from different witnesses.
In local encoding schemes, the value of the wit attribute can be enforced as IDREFS, such that only witnesses referred to in a <witList> element may occur as witnesses to a reading.

Remarks:
This element class defines attributes inherited by <rdg>, <lem>, and <rdg.grp>.

Part: additional tag set for text criticism
Member of classes:
DTD file: teitc2.ent
Occurs within: [none]

Members of: lem rdg rdgGrp restore

```
<!ENTITY % a.readings '      %a.global
          wit                CDATA              #IMPLIED
          type               CDATA              #IMPLIED
          cause              CDATA              #IMPLIED
          varSeq             NUMBER             #IMPLIED
          resp               CDATA              %INHERITED

          hand               IDREF              %INHERITED'    >
```

Discussed in: 19.1 ('The Apparatus Entry, Readings, and Witnesses') on p. 469

Class Name: refsys (i.e. reference system elements) refsys

Description: milestone-style elements used in reference systems
Attributes: [None: global and inherited attributes only.]

Remarks:

Part: base tag set for common core features
Member of classes:
DTD file: teiclas2.ent
Occurs within: [none]
Members of: cb lb milestone pb

```
<!ENTITY % m.refsys '
  %x.refsys pb | milestone | lb | cb'                          >
```

Discussed in: 3.7 ('Element Classes') on p. 51; 6.9.3 ('Milestone Tags') on p. 158

Class Name: seg seg

Description: elements for arbitrary segmentation.
Attributes:

type characterizes the type of segment.
Data type: CDATA
Value: For a <cl> may take values such as finite, nonfinite, declarative, interrogative, relative etc. For a <phr> or <w>, values such as noun, verb, preposition, etc., may be used. For an <m> element, values such as clitic, prefix, stem will be more appropriate. For a <c> element, values such as letter, punctuation, digit may be used.
Default: #IMPLIED

function characterizes the function of the segment.
Data type: CDATA
Value: For a <cl>, may take values such as coordinate, subject, adverbial etc. For a <phr>, such values as subject, predicate etc. may be more appropriate.
Default: #IMPLIED

Remarks:
The principles on which segmentation is carried out, and any special codes or attribute values used, should be defined explicitly in the <segmentation> element of the <encodingDesc> within the associated TEI header.

Part: base tag set for common core features
Member of classes: phrase
DTD file: teiclas2.ent
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q

quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

Members of: anchor c cl m phr s seg w

```
<!ENTITY % m.seg '
  %x.seg w | seg | s | phr | m | cl | c | anchor'              >
<!ENTITY % a.seg '           %a.global
          type               CDATA              #IMPLIED
          function           CDATA              #IMPLIED'     >
```

Discussed in: 14.3 ('Segments and Anchors') on p. 355; 15.1 ('Linguistic Segment Categories') on p. 382

sgmlKeywords

Class Name: sgmlKeywords (i.e. SGML keywords)

Description: elements whose content is an SGML identifier or tag of some sort (generic identifier of an element type, name of an attribute, etc.).

Attributes: [None: global and inherited attributes only.]

Remarks:
These elements are defined in the auxiliary tag set for tag set documentation; they may be useful in writing SGML documentation as well.

Part: auxiliary tag set for tag set declarations

Member of classes: phrase

DTD file: teiclas2.ent

Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

Members of: att gi tag val

```
<!ENTITY % m.sgmlKeywords '
  %x.sgmlKeywords val | tag | gi | att'                        >
```

Discussed in: 27 ('Tag Set Documentation') on p. 601

singleVal

Class Name: singleVal (i.e. single values)

Description: elements which express single feature values in feature structures.

Attributes: [None: global and inherited attributes only.]

Part: additional tag set for feature structures

Member of classes: featureVal

DTD file: teifsd2

Occurs within: if vDefault vRange

Members of: dft msr nbr rate str sym uncertain *boolean* (any none) *binary* (minus plus)

```
<!ENTITY % m.singleVal '
  %x.singleVal %m.boolean | %m.binary | uncertain | sym | str |
  rate | nbr | msr | dft'                                      >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

stageDirection

Class Name: stageDirection (i.e. stage directions)

Description: groups elements for specialized stage directions defined in the additional tag set for performance texts.

Attributes: [None: global and inherited attributes only.]

Remarks:
Stage directions are members of class *inter*: that is, they can appear between or within component-level elements.

Part: additional tag set for performance texts

Member of classes: comp.drama, inter

DTD file: `teidram2.ent`

Occurs within: add admin argument body camera caption case castList cell colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref

Members of: camera caption move sound tech view

```
<!ENTITY % m.stageDirection '
  %x.stageDirection view | tech | sound | move | caption |
  camera'                                                  >
```

Discussed in: 10.3 ('Other Types of Performance Text') on p. 246

Class Name: TEIform (i.e. TEI name form)

TEIform

Description: defines an attribute (TEIform) common to all tags in the TEI scheme, and recommended for all user-defined extensions.

Attributes:

TEIform (i.e. TEI form of generic identifier) indicates the standard TEI name (generic identifier) for a given element.
Data type: NAME
Value: must be a valid SGML name; the default is specified in
Default: #IMPLIED
Example:

```
<fn TEIform=note>This is a footnote; its tag uses a non-standard
 name defined by the user; the attribute TEIform indicates that
 the normal TEI name for the element is NOTE.</fn>
```

In the TEI DTDs, the default value for this attribute is always the same as the generic identifier of the element. If an element is renamed using the techniques described in chapter 29 ('Modifying the TEI DTD') on p. 619, the attribute declaration for TEIform will be left undisturbed; the default value will thus still be the standard TEI name for the element. TEI-aware application programs can thus process TEI-conformant documents which rename TEI elements, since by consulting the TEIform attribute value the application can learn the standard name for the element and process it accordingly.
In the normal course of events, this attribute will never be specified in a TEI-conformant document; all occurrences will have the default value. In some special circumstances, it can be useful to specify a non-default value on some instances of an element; this allows application programs to process correctly a locally defined element which usually corresponds to one TEI element (which would be expressed by the default value) but sometimes to another TEI element (which would be expressed by explicit values attached to the element instance).

Remarks:
The attribute TEIform, though common to all tags in the TEI encoding scheme, is not defined as part of the *global* class for technical reasons. Since its default value must be specified separately for each element type, no elements actually inherit the attribute from the element class *TEIform*; each defines the attribute separately.

Part: base tag set for common core features

Member of classes:
DTD file:
Occurs within: [none]
Members of:

```
<!ENTITY % a.TEIform '       %a.global
          TEIform            NAME                 #IMPLIED'      >
```

Discussed in: 3.5 ('Global Attributes') on p. 42

temporalExpr

Class Name: temporalExpr (i.e. temporal expression)

Description: groups component elements of temporal expressions involving dates and time.
Attributes:

value supplies the value of a date or time in a standard form.
Data type: CDATA
Value: Any string representing a temporal expression in standard format; recommended form is "yyyy-mm-dd", as defined by ISO 8601: 1988, *Data elements and interchange formats — Information interchange — Representation of dates and times.*
Default: #IMPLIED
The standard form used should be described in the <stdVals> element in the TEI header. Standard forms may be defined from scratch, or borrowed from existing practice.

type provides any application-, linguistic- or culture-specific classication for the component.
Data type: CDATA
Value: Any string of characters.
Default: #IMPLIED
Example:

```
<dateStruct value='14-05-1993'>
 <day type=nominal>Friday</day>,
 <day type=numeric>14</day>
 <month type=nominal>May</month>
 <year type=nominal value='1993'>
   in the year of our Lord One Thousand
   Nine Hundred and Ninety Three
 </year>
</dateStruct>
```

reg (i.e. regularization) gives a normalized or regularized form of the temporal expression.
Data type: CDATA
Value: Any string of characters.
Default: #IMPLIED
In providing a "regularized" form, no claim is made that the form in the source text is incorrect; the regularized form is simply that chosen as the main form for purposes of unifying variant forms under a single heading.
Example:

```
<dateStruct value='09-06-1807'>
  <month>June</month> <day reg='9'>9th</day>
</dateStruct>:
The period is approaching which will terminate my
present copartnership. On the
  <dateStruct value='01-01-1808'>
     <day reg='1'>1st</day>
     <month reg='January'>Jany.</month>
   </dateStruct>
next, it expires by its own limitation.

At <time><event reg='evening'>sunset</event></time> we walked
walked to the beach.
```

full indicates whether the date element is given in full, as an abbreviation or simply as an initial

Data type: (YES | ABB | INIT)
Sample values include:

yes the name component is spelled out in full.
abb the name component is given in an abbreviated form.
init the name component is indicated only by one initial.

Default: YES

Part: additional tag set for names and dates
Member of classes:
DTD file: teind2.ent
Occurs within: dateStruct timeStruct
Members of: dateStruct day distance hour minute month occasion offset second timeStruct week year

```
<!ENTITY % a.temporalExpr '  %a.global
          value              CDATA              #IMPLIED
          type               CDATA              #IMPLIED
          reg                CDATA              #IMPLIED
          full               (yes | abb | init) yes'          >
```

Discussed in: 20.4 ('Dates and Time') on p. 499

Class Name: terminology (i.e. global attributes for terminological data.) terminology

Description: default declaration for class *terminology*: when the base tag set for terminological data is not selected, no attributes are defined for this class.
Attributes: [None: global and inherited attributes only.]
Part: base tag set for terminological data
Member of classes:
DTD file: teiclas2.ent
Occurs within: [none]
Members of:

```
<!ENTITY % a.terminology '   %a.global'                                    >
```

Discussed in: 3.5 ('Global Attributes') on p. 42

Class Name: terminology (i.e. global attributes for terminological data.) terminology

Description: defines attributes for all elements in documents which use the base tag set for terminological data.
Attributes:

group indicates the group (term and related elements) to which this element should be associated by specifying a string matching the n attribute value on an appropriate element.
Data type: CDATA
Value: any string matching the n attribute value on the <term> element to which the group is attached.
Default: #IMPLIED
The group attribute provides a specialized pointing mechanism for use within <termEntry> elements.

grpPtr indicates the group (term and related elements) to which this element should be associated by specifying its unique identifier, where this is available.
Data type: IDREF
Value: the value specified must match a value supplied as the value for an id attribute on some <term> element in the current SGML document.
Default: #IMPLIED
The group attribute provides a specialized pointing mechanism for use within <termEntry> elements.

depend indicates the parent element to which this element should be associated by specifying a string matching the n attribute value on an appropriate element.
Data type: CDATA
Value: any string matching the n attribute value on the element to which the dependent element is attached.

Default: #IMPLIED

depPtr indicates the parent element to which this element should be associated by specifying its unique identifier, where this is available.
Data type: IDREF
Value: the value specified must match a value supplied as the value for an id attribute on some <term> element in the current SGML document.
Default: #IMPLIED

Remarks:
The attributes shared by this element class are used for linking elements, possibly not adjacent in the record, which are related (e.g. a grammatical annotation and the term it describes). If no attribute is specified, the element is assumed to relate to the most recently specified <term> or <otherForm> element.

Part: base tag set for terminological data
Member of classes:
DTD file: teiterm2.ent
Occurs within: [none]
Members of: *global* ()

```
<!ENTITY % a.terminology '
          group              CDATA              #IMPLIED
          grpPtr             IDREF              #IMPLIED
          depend             CDATA              #IMPLIED
          depPtr             IDREF              #IMPLIED'      >
```

Discussed in: 13.2 ('Tags for Terminological Data') on p. 312; 13.3.3 ('Flat Term Entries Using Group and Depend Attributes') on p. 317; 13.4 ('Overall Structure of Terminological Documents') on p. 319

terminologyInclusions

Class Name: terminologyInclusions

Description: elements which may be included at any point within a terminology entry.
Attributes: [None: global and inherited attributes only.]
Part: base tag set for terminological data
Member of classes:
DTD file: teiterm2.ent
Occurs within: termEntry [May appear anywhere within: termEntry]
Members of: date dateStruct note ptr ref xptr xref

```
<!ENTITY % m.terminologyInclusions '
  %x.terminologyInclusions xref | xptr | ref | ptr | note |
  dateStruct | date'                                          >
```

Discussed in: 13 ('Terminological Databases') on p. 311

terminologyMisc

Class Name: terminologyMisc (i.e. miscellaneous terminology-data elements)

Description: elements which can appear together at various points in terminological entries.
Attributes: [None: global and inherited attributes only.]
Part: base tag set for terminological data
Member of classes:
DTD file: teiterm2.ent
Occurs within: ofig termEntry tig
Members of: admin descrip

```
<!ENTITY % m.terminologyMisc '
  %x.terminologyMisc descrip | admin'                         >
```

Discussed in: 13 ('Terminological Databases') on p. 311

timed

Class Name: timed

Description: elements which have a duration in time expressed either absolutely or by reference to an alignment map.
Attributes:

start indicates the location within a temporal alignment at which this element begins.
Data type: IDREF
Value: contains the identifier of a previously defined <loc> element
Default: #IMPLIED
If no value is supplied, the element is assumed to follow the immediately preceding element at the same hierarchic level.

end indicates the location within a temporal alignment at which this element ends.
Data type: IDREF
Value: contains the identifier of a previously defined <loc> element
Default: #IMPLIED
If no value is supplied, the element is assumed to precede the immediately following element at the same hierarchic level.

dur (i.e. duration) indicates the length of this element in time, using either specific units or the units specified on the associated temporal alignment.
Data type: CDATA
Value: contains a number optionally followed by a standard unit indicator
Default: #IMPLIED
If units are not defaulted, they should be represented using standard abbreviations (s for second, m for minute, etc.)

Part: base tag set for spoken materials
Member of classes:
DTD file: teispok2.ent
Occurs within: [none]
Members of: event kinesic pause u vocal

```
<!ENTITY % a.timed '          %a.global
          start               IDREF               #IMPLIED
          end                 IDREF               #IMPLIED
          dur                 CDATA               #IMPLIED'      >
```

Discussed in: 11.2.5 ('Temporal Information') on p. 257

Class Name: tpParts (i.e. Title page elements) tpParts

Description: This class groups elements which can occur as direct constituents of a title page (<docTitle>, <docAuth>, <docImprint>, <epigraph>, etc.)
Attributes: [None: global and inherited attributes only.]
Part: base tag set for common core features
Member of classes:
DTD file: teifron2
Occurs within: titlePage
Members of: byline docAuthor docDate docEdition docImprint docTitle epigraph imprimatur titlePart

```
<!ENTITY % m.tpParts '
  %x.tpParts titlePart | imprimatur | epigraph | docTitle |
  docImprint | docEdition | docDate | docAuthor | byline'       >
```

Discussed in: 7.5 ('Title Pages') on p. 203

Class Name: xPointer (i.e. extended-pointer elements) xPointer

Description: groups elements which use the TEI extended pointer mechanism to point at locations which have no SGML ID.
Attributes:

doc (i.e. document or file) specifies the document within which the desired location is to be found.
Data type: ENTITY
Value: The name of a system entity associated with the document within which the target of this extended pointer is to be found, by default the current document.
Default: #IMPLIED
Example:

```
<xptr doc='Chap2' from='id (e23)'>
```

The system entity named by this attribute must be declared (as a SUBDOC entity or a non-SGML entity) in the DTD of the document containing the extended pointer.

from specifies the start of the destination of the pointer, as an expression in the TEI extended-pointer notation described in section 14.2 ('Extended Pointers') on p. 340.
Data type: %EXTPTR;
Value: The value specified must be a valid expression in the TEI extended pointer notation defined in section 14.2 ('Extended Pointers') on p. 340.
Default: "ROOT"
If no value is specified, the target is the whole of the document identified by the doc attribute.

to specifies the endpoint of the destination of the pointer, as an expression in the TEI extended pointer notation.
Data type: %EXTPTR;
Value: The value specified must be a valid expression in the TEI extended pointer notation defined in section 14.2 ('Extended Pointers') on p. 340.
Default: "DITTO"
Example:

```
<xptr doc=OrbisPictus from='id (animalia)'
      to='id (aquaticae)'>
```

This attribute may only be supplied if the from attribute is also supplied, in which case the destination is defined to extend from the beginning of the location specified by the from attribute, up to the end of that specified by the to attribute. It is an error for the to attribute to specify a location whose end precedes the beginning of the location specified by from; it is not an error for the scopes to overlap.
If no value is specified, the target is the location specified by the from attribute.

Remarks:
This class belongs to the larger class *pointer*, which means its elements also inherit the attributes of that class.

Part: additional tag set for
Member of classes: pointer
DTD file: teiclas2.ent
Occurs within: [none]
Members of: xptr xref

```
<!ENTITY % a.xPointer '      %a.global
                             %a.pointer
          doc                ENTITY              #IMPLIED
          from               %extPtr;            "ROOT"
          to                 %extPtr;            "DITTO"'      >
```

Discussed in: 14.2 ('Extended Pointers') on p. 340

Chapter 34

Entities

Entity Name: component (i.e. component for dictionaries) component

Description: defines the set of component-level elements for dictionaries; these are elements which can appear directly within text bodies or text divisions.
DTD file: teiclas2.ent
Referred to by: component.seq
Expansion: '(%m.common | %m.comp.dictionaries)'
Discussed in: 3.7 ('Element Classes') on p. 51

Entity Name: component (i.e. component for drama) component

Description: defines the set of component-level elements for drama; these are elements which can appear directly within text bodies or text divisions.
DTD file: teiclas2.ent
Referred to by: component.seq
Expansion: '(%m.common | %m.comp.drama)'
Discussed in: 3.7 ('Element Classes') on p. 51

Entity Name: component (i.e. component for mixed or general base) component

Description: defines the set of component-level elements for use with the mixed or general base; these are elements which can appear directly within text bodies or text divisions.
DTD file: teiclas2.ent
Referred to by: component.seq
Expansion: '(%m.common %mix.verse %mix.drama %mix.spoken %mix.dictionaries %mix.terminology)'
Discussed in: 3.7 ('Element Classes') on p. 51

Entity Name: component (i.e. component for prose) component

Description: defines the set of component-level elements for prose; these are elements which can appear directly within text bodies or text divisions.
DTD file: teiclas2.ent
Referred to by: component.seq
Expansion: '(%m.common)'
Discussed in: 3.7 ('Element Classes') on p. 51

Entity Name: component (i.e. component for transcription of spoken texts) component

Description: defines the set of component-level elements for spoken texts; these are elements which can appear directly within text bodies or text divisions.
DTD file: teiclas2.ent
Referred to by: component.seq
Expansion: '(%m.common | %m.comp.spoken)'
Discussed in: 3.7 ('Element Classes') on p. 51

component

Entity Name: component (i.e. component for terminology)

Description: defines the set of component-level elements for terminology; these are elements which can appear directly within text bodies or text divisions.

DTD file: teiclas2.ent

Referred to by: component.seq

Expansion: '(%m.common | %m.comp.terminology)'

Discussed in: 3.7 ('Element Classes') on p. 51

component

Entity Name: component (i.e. component for verse)

Description: defines the set of component-level elements for verse; these are elements which can appear directly within text bodies or text divisions.

DTD file: teiclas2.ent

Referred to by: component.seq

Expansion: '(%m.common | %m.comp.verse)'

Discussed in: 3.7 ('Element Classes') on p. 51

component.plus

Entity Name: component.plus (i.e. component sequence)

Description: defines a sequence of components as needed in the general base tag set, allowing components from any base to be used, but preventing their mixing.

Remarks:
When used in a content model, this entity requires at least one component-level element to occur.

DTD file: teiclas2.ent

Referred to by:

Expansion: '(%gen.verse %gen.drama %gen.spoken %gen.dictionaries %gen.terminology TEI...end) | ((%m.common)+, (%gen.verse %gen.drama %gen.spoken %gen.dictionaries %gen.terminology TEI...end)?'

Discussed in: 3.7 ('Element Classes') on p. 51

component.seq

Entity Name: component.seq (i.e. component sequence for general combined base)

Description: defines a sequence of components as needed in the general base tag set, allowing components from any base to be used, but preventing their mixing.

Remarks:
When used in a content model, this entity, like other entities with names of the form "x.seq", allows zero or more occurrences of its content to occur.

DTD file: teiclas2.ent

Referred to by: specialPara

Expansion: '(%m.common)*, (%gen.verse %gen.drama %gen.spoken %gen.dictionaries %gen.terminology TEI...end)?'

Discussed in: 3.7 ('Element Classes') on p. 51

component.seq

Entity Name: component.seq (i.e. component-sequence)

Description: defines a sequence of component-level elements (such as paragraphs or lists) which can occur directly within text divisions and in similar positions.

Remarks:
This parameter entity is used in each base tag set to define the content of and similar elements.

DTD file: teiclas2.ent

Referred to by: specialPara

Expansion: '(%component;)*'

Discussed in: 3.7 ('Element Classes') on p. 51

extPtr

Entity Name: extPtr (i.e. extended-pointer expression)

Description: used as the declared value of an attribute, indicates that all values of that attribute must be valid expressions in the TEI extended pointer notation defined in section

14.2.1 (‘Extended Pointer Elements’) on p. 340.

Remarks:

DTD file: teikey2.ent
Referred to by:
Expansion: ’CDATA’
Discussed in: 14.2.1 (‘Extended Pointer Elements’) on p. 340

Entity Name: formulaContent formulaContent

Description: defines the content model for the <formula> element.
DTD file: teifig2.ent
Referred to by:
Expansion: ’CDATA’
Discussed in: 22.2 (‘Formulae’) on p. 527

Entity Name: formulaNotations formulaNotations

Description: specifies the set of notations which may be used for the <formula> element.

Remarks:

This will normally be defined either as CDATA (the default), or as a string such as NOTATION (tex | eqn)

DTD file: teifig2.ent
Referred to by:
Expansion: ’CDATA’
Discussed in: 22.2 (‘Formulae’) on p. 527

Entity Name: gen.dictionaries (i.e. dictionary part of general-base component sequence) gen.dictionaries

Description: contains a string used in constructing the definition of component sequence used in the general base tag set.
DTD file: teiclas2.ent
Referred to by: component.seq component.plus
Expansion: ’((%m.comp.dictionaries), (%m.common | %m.comp.dictionaries)*) |’
Discussed in: 3.7 (‘Element Classes’) on p. 51

Entity Name: gen.dictionaries (i.e. dictionary part of general-base component sequence) gen.dictionaries

Description: contains a string used in constructing the definition of component sequence used in the general base tag set.

Remarks:

This version of this entity is used if the dictionary base is not selected.

DTD file: teiclas2.ent
Referred to by: component.seq component.plus
Expansion: ”
Discussed in: 3.7 (‘Element Classes’) on p. 51

Entity Name: gen.drama (i.e. drama part of general-base component sequence) gen.drama

Description: contains a string used in constructing the definition of component sequence used in the general base tag set.
DTD file: teiclas2.ent
Referred to by: component.seq component.plus
Expansion: ’((%m.comp.drama), (%m.common | %m.comp.drama)*) |’
Discussed in: 3.7 (‘Element Classes’) on p. 51

Entity Name: gen.drama (i.e. drama part of general-base component sequence) gen.drama

Description: contains a string used in constructing the definition of component sequence used in the general base tag set.

Remarks:

This version of this entity is used if the drama base is not selected.

DTD file: teiclas2.ent
Referred to by: component.seq component.plus
Expansion: ''
Discussed in: 3.7 ('Element Classes') on p. 51

gen.spoken

Entity Name: gen.spoken (i.e. spoken-text part of general-base component sequence)
Description: contains a string used in constructing the definition of component sequence used in the general base tag set.
DTD file: teiclas2.ent
Referred to by: component.seq component.plus
Expansion: '((%m.comp.spoken), (%m.common | %m.comp.spoken)*) |'
Discussed in: 3.7 ('Element Classes') on p. 51

gen.spoken

Entity Name: gen.spoken (i.e. spoken-text part of general-base component sequence)
Description: contains a string used in constructing the definition of component sequence used in the general base tag set.
Remarks:
This version of this entity is used if the spoken base is not selected.
DTD file: teiclas2.ent
Referred to by: component.seq component.plus
Expansion: ''
Discussed in: 3.7 ('Element Classes') on p. 51

gen.terminology

Entity Name: gen.terminology (i.e. terminology part of general-base component sequence)
Description: contains a string used in constructing the definition of component sequence used in the general base tag set.
DTD file: teiclas2.ent
Referred to by: component.seq component.plus
Expansion: '((%m.comp.terminology), (%m.common | %m.comp.terminology)*) |'
Discussed in: 3.7 ('Element Classes') on p. 51

gen.terminology

Entity Name: gen.terminology (i.e. terminology part of general-base component sequence)
Description: contains a string used in constructing the definition of component sequence used in the general base tag set.
Remarks:
This version of this entity is used if the terminology base is not selected.
DTD file: teiclas2.ent
Referred to by: component.seq component.plus
Expansion: ''
Discussed in: 3.7 ('Element Classes') on p. 51

gen.verse

Entity Name: gen.verse (i.e. verse part of general-base component sequence)
Description: contains a string used in constructing the definition of component sequence used in the general base tag set.
DTD file: teiclas2.ent
Referred to by: component.seq component.plus
Expansion: '((%m.comp.verse), (%m.common | %m.comp.verse)*) |'
Discussed in: 3.7 ('Element Classes') on p. 51

gen.verse

Entity Name: gen.verse (i.e. verse part of general-base component sequence)
Description: contains a string used in constructing the definition of component sequence used in the general base tag set.
Remarks:
This version of this entity is used if the verse base is not selected.
DTD file: teiclas2.ent

Referred to by: component.seq component.plus
Expansion: ”
Discussed in: 3.7 (‘Element Classes’) on p. 51

Entity Name: INHERITED INHERITED

Description: as a default attribute value, indicates that if not specified, the attribute value is inherited from corresponding attribute of the parent element.

Remarks:
For the SGML parser, this entity has the same effect as specifying a default value as `IMPLIED`. For the user and the application program, however, `INHERITED` has a different effect, specifying as it does exactly how the application program is to infer the correct default value: the value is to be inherited from the attribute of the same name on the immediately enclosing element. If that element, too, specifies no value, then its value will have been inherited from its own immediately enclosing element, etc. (If the attribute is not declared for all elements, the value is inherited from the nearest ancestor for which the attribute is declared.)
If no ancestor element has a value specified for the attribute, the value is undefined. Encoders are encouraged to provide an explicit value for inherited attributes on the outermost elements for which they are declared; it is, however, not an error for the outermost element to specify no attribute value for an attribute with a default of `%INHERITED`.
The most prominent example of attribute value inheritance is the TEI global attribute lang.

DTD file: `teiwsd2`
Referred to by:
Expansion: ’#IMPLIED’
Discussed in: 3.7 (‘Element Classes’) on p. 51

Entity Name: ISO-date ISO-date

Description: as the declared value of an attribute, indicates that the attribute value should be a legal ISO date in the format defined by ISO 714[?], viz. yyyy-mm-dd.

Remarks:
For the SGML parser, this entity has the same effect as specifying a declared value of “CDATA”. For the user and the application program, however, “%ISO-date” documents a restriction which SGML is not currently capable of enforcing.
The most prominent examples of this declared value type are the value attribute of the <date> element in the core tag set, and the date attribute of the <admin> element in the base tag set for terminology.

DTD file: `teiwsd2`
Referred to by:
Expansion: ’CDATA’
Discussed in: 3.8.3 (‘Parameter Entities for TEI Keywords’) on p. 67

Entity Name: mix.dictionaries (i.e. mixed-base dictionary components) mix.dictionaries

Description: contains a string used in constructing the definition of *component* used in the mixed base tag set.
DTD file: `teidict2.ent`
Referred to by: component
Expansion: ’| %m.comp.dictionaries’
Discussed in: 3.7 (‘Element Classes’) on p. 51

Entity Name: mix.dictionaries (i.e. mixed-base dictionaries components) mix.dictionaries

Description: default declaration of a string used in constructing the definition of *component* used in the mixed base tag set.
DTD file: `teiclas2.ent`
Referred to by: component

Expansion: ”
Discussed in: 3.7.8 (‘Components in Mixed and General Bases’) on p. 62

mix.drama

Entity Name: mix.drama (i.e. mixed-base drama components)
Description: contains a string used in constructing the definition of *component* used in the mixed base tag set.
DTD file: teidram2.ent
Referred to by: component
Expansion: ’| %m.comp.drama’
Discussed in: 3.7 (‘Element Classes’) on p. 51

mix.drama

Entity Name: mix.drama (i.e. mixed-base drama components)
Description: default declaration of a string used in constructing the definition of *component* used in the mixed base tag set.
DTD file: teiclas2.ent
Referred to by: component
Expansion: ”
Discussed in: 3.7.8 (‘Components in Mixed and General Bases’) on p. 62

mix.spoken

Entity Name: mix.spoken (i.e. mixed-base spoken-text components)
Description: contains a string used in constructing the definition of *component* used in the mixed base tag set.
DTD file: teispok2.ent
Referred to by: component
Expansion: ’| %m.comp.spoken’
Discussed in: 3.7 (‘Element Classes’) on p. 51

mix.spoken

Entity Name: mix.spoken (i.e. mixed-base spoken components)
Description: default declaration of a string used in constructing the definition of *component* used in the mixed base tag set.
DTD file: teiclas2.ent
Referred to by: component
Expansion: ”
Discussed in: 3.7.8 (‘Components in Mixed and General Bases’) on p. 62

mix.terminology

Entity Name: mix.terminology (i.e. mixed-base terminology components)
Description: contains a string used in constructing the definition of *component* used in the mixed base tag set.
DTD file: teiterm2.ent
Referred to by: component
Expansion: ’| %m.comp.terminology’
Discussed in: 3.7 (‘Element Classes’) on p. 51

mix.terminology

Entity Name: mix.terminology (i.e. mixed-base terminology components)
Description: default declaration of a string used in constructing the definition of *component* used in the mixed base tag set.
DTD file: teiclas2.ent
Referred to by: component
Expansion: ”
Discussed in: 3.7.8 (‘Components in Mixed and General Bases’) on p. 62

mix.verse

Entity Name: mix.verse (i.e. mixed-base verse components)
Description: contains a string used in constructing the definition of *component* used in the mixed base tag set.
DTD file: teivers2.ent

Referred to by: component
Expansion: '| %m.comp.verse'
Discussed in: 3.7 ('Element Classes') on p. 51

Entity Name: mix.verse (i.e. mixed-base verse components) mix.verse

Description: default declaration of a string used in constructing the definition of *component* used in the mixed base tag set.
DTD file: teiclas2.ent
Referred to by: component
Expansion: ''
Discussed in: 3.7.8 ('Components in Mixed and General Bases') on p. 62

Entity Name: paraContent (i.e. paragraph content) paraContent

Description: defines the legal version for paragraphs and similar elements.
DTD file: teiclas2.ent
Referred to by: specialPara
Expansion: '(#PCDATA | %m.phrase | %m.inter)*'
Discussed in: 3.7 ('Element Classes') on p. 51

Entity Name: phrase phrase

Description: defines a phrase as character data or any phrase-level element.

Remarks:
This entity is used in the declaration of *phrase.seq*.

DTD file: teiclas2.ent
Referred to by: phrase.seq
Expansion: '(#PCDATA | %m.phrase)'
Discussed in: 3.7 ('Element Classes') on p. 51

Entity Name: phrase.seq (i.e. phrase sequence) phrase.seq

Description: defines a sequence of character data and phrase-level elements.
DTD file: teiclas2.ent
Referred to by:
Expansion: '(%phrase;)*'
Discussed in: 3.7.7 ('Standard Content Models') on p. 61

Entity Name: seq (i.e. sequence) seq

Description: defines a sequence of elements (such as paragraphs) which can occur directly within text divisions and in similar positions.

Remarks:
This parameter entity is used in each base tag set to define the content of and other elements.

DTD file: teiterm2.ent
Referred to by:
Expansion: '(%m.common; | %m.comp.terminology;)* '
Discussed in: 13.4 ('Overall Structure of Terminological Documents') on p. 319

Entity Name: specialPara (i.e. 'special' paragraph content) specialPara

Description: defines the content model of elements such as notes or list items, which either contain a series of component-level elements or else have the same structure as a paragraph, containing a series of phrase-level and inter-level elements.
DTD file: teiclas2.ent
Referred to by:
Expansion: '(((%m.chunk), (%component.seq)) | (%paraContent))'
Discussed in: 3.7 ('Element Classes') on p. 51

TEI.analysis.dtd

Entity Name: TEI.analysis.dtd (i.e. TEI analysis-base DTD)
Description: identifies the file containing element and attribute list declarations for the base tag set for simple analysis.
Remarks:
This entity is declared with this value when the user includes the base tag set for simple analysis.
DTD file: `tei2`
Referred to by:
Expansion: system 'teiana2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

TEI.analysis.ent

Entity Name: TEI.analysis.ent (i.e. TEI analysis-base entities)
Description: identifies the file containing parameter entity declarations for the element classes defined in the base tag set for simple analysis.
Remarks:
This entity is declared with this value when the user includes the base tag set for simple analysis.
DTD file: `teiclas2.ent`
Referred to by:
Expansion: system 'teiana2.ent'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

TEI.analysis

Entity Name: TEI.analysis (i.e. TEI simple analytic mechanisms DTD fragment)
Description: controls the inclusion, in the DTD, of element and attribute declarations for simple analytic mechanisms.
Remarks:
To include elements and attributes for simple analytic mechanisms in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.
DTD file: `teikey2.ent`
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.6 ('The TEI2.DTD File') on p. 45

TEI.back.dtd

Entity Name: TEI.back.dtd
Description: defines the file in which tags for back matter are defined.
Remarks:
This parameter entity is used in each base tag set to include the back-matter tags, which are common to all bases.
DTD file: `teistr2`
Referred to by:
Expansion: system 'teiback2.dtd'
Discussed in: 7.6 ('Back Matter') on p. 205

TEI.certainty.dtd

Entity Name: TEI.certainty.dtd (i.e. TEI certainty-base DTD)
Description: identifies the file containing element and attribute list declarations for the base tag set for certainty and uncertainty.
Remarks:
This entity is declared with this value when the user includes the base tag set for certainty and uncertainty.
DTD file: `tei2`
Referred to by:
Expansion: system 'teicert2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

TEI.certainty

Entity Name: TEI.certainty (i.e. TEI certainty DTD fragment)

Description: controls the inclusion, in the DTD, of element and attribute declarations for certainty.

Remarks:
To include elements and attributes for certainty in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: `teikey2.ent`
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.5 ('Global Attributes') on p. 42; 3.6 ('The TEI2.DTD File') on p. 45

Entity Name: TEI.core.dtd (i.e. TEI core) TEI.core.dtd

Description: identifies the file containing element and attribute list declarations for the TEI core elements.

Remarks:
This entity is included in all TEI DTDs.

DTD file: `teitsd2`
Referred to by:
Expansion: system 'teicore2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

Entity Name: TEI.corpus.dtd (i.e. TEI corpus-base DTD) TEI.corpus.dtd

Description: identifies the file containing element and attribute list declarations for the base tag set for corpora.

Remarks:
This entity is declared with this value when the user includes the base tag set for corpora.

DTD file: `tei2`
Referred to by:
Expansion: system 'teicorp2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

Entity Name: TEI.corpus (i.e. TEI corpora DTD fragment) TEI.corpus

Description: controls the inclusion, in the DTD, of element and attribute declarations for the description of corpora.

Remarks:
To include elements and attributes for corpora in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: `teikey2.ent`
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.5 ('Global Attributes') on p. 42; 3.6 ('The TEI2.DTD File') on p. 45

Entity Name: TEI.dictionaries.dtd (i.e. TEI dictionary-base DTD) TEI.dictionaries.dtd

Description: identifies the file containing element and attribute list declarations for the base tag set for dictionaries.

Remarks:
This entity is declared with this value when the user includes the base tag set for dictionaries.

DTD file: `tei2`
Referred to by:
Expansion: system 'teidict2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

Entity Name: TEI.dictionaries.ent (i.e. TEI dictionary-base entities) TEI.dictionaries.ent

Description: identifies the file containing parameter entity declarations for the element classes defined in the base tag set for dictionaries.

Remarks:
This entity is declared with this value when the user includes the base tag set for dictionaries.

DTD file: teiclas2.ent
Referred to by:
Expansion: system 'teidict2.ent'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

TEI.dictionaries

Entity Name: TEI.dictionaries (i.e. TEI dictionaries DTD fragment)
Description: controls the inclusion, in the DTD, of element and attribute declarations for dictionaries.

Remarks:
To include elements and attributes for dictionaries in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: teikey2.ent
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.5 ('Global Attributes') on p. 42; 3.6 ('The TEI2.DTD File') on p. 45

TEI.drama.dtd

Entity Name: TEI.drama.dtd (i.e. TEI drama-base DTD)
Description: identifies the file containing element and attribute list declarations for the base tag set for drama.

Remarks:
This entity is declared with this value when the user includes the base tag set for drama.

DTD file: tei2
Referred to by:
Expansion: system 'teidram2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

TEI.drama.ent

Entity Name: TEI.drama.ent (i.e. TEI drama-base entities)
Description: identifies the file containing parameter entity declarations for the element classes defined in the base tag set for drama.

Remarks:
This entity is declared with this value when the user includes the base tag set for drama.

DTD file: teiclas2.ent
Referred to by:
Expansion: system 'teidram2.ent'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

TEI.drama

Entity Name: TEI.drama (i.e. TEI drama DTD fragment)
Description: controls the inclusion, in the DTD, of element and attribute declarations for drama.

Remarks:
To include elements and attributes for drama in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: teikey2.ent
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.5 ('Global Attributes') on p. 42; 3.6 ('The TEI2.DTD File') on p. 45

TEI.elementClasses

Entity Name: TEI.elementClasses
Description: file containing the parameter entity declarations which define element classes for content models, attributes shared among elements of a class, and global attributes.

Remarks:

DTD file: teitsd2
Referred to by:
Expansion: system 'teiclas2.ent'
Discussed in: 3.6 ('The TEI2.DTD File') on p. 45; 3.7 ('Element Classes') on p. 51

Entity Name: TEI.elementNames TEI.elementNames

Description: file containing parameter entity declarations for all generic identifiers of the encoding scheme.

Remarks:
The parameter entities in this file all take the same form as the two shown below:

```
<!ENTITY % n.div0 'div0' >
<!ENTITY % n.div1 'div1' >
```

Element and attribute-list declarations in the DTDs refer to the parameter entity *n.div1*, not directly to the generic identifier *div1*. As a result, the declarations will function as desired even if a new generic identifier is substituted (e.g. "caput" for "div1" and "liber" for "div0":

```
<!ENTITY % n.div0 'liber' >
<!ENTITY % n.div1 'caput' >
```

This allows generic identifiers to be renamed conveniently, to provide names in languages other than English, or to provide shorter names than those documented here. See further chapter 29 ('Modifying the TEI DTD') on p. 619.

DTD file: teitsd2
Referred to by:
Expansion: system 'teigis2.ent'
Discussed in: 3.8.2 ('Parameter Entities for Element Generic Identifiers') on p. 66

Entity Name: TEI.extensions.dtd TEI.extensions.dtd

Description: file (if any) containing local modifications to the TEI DTDs.

Remarks:
This entity is embedded in the TEI DTDs after the TEI element classes are embedded, and immediately before the base tag set and additional tag sets selected by the user are embedded. By default, the entity expands to the empty string; the user can override this default by declaring the entity with an appropriate value, typically this will take the form

```
<!ENTITY % TEI.extensions.dtd system 'project.dtd' >
```

DTD file: tei2
Referred to by:
Expansion: "
Discussed in: 3.6 ('The TEI2.DTD File') on p. 45

Entity Name: TEI.extensions.ent TEI.extensions.ent

Description: file (if any) containing local modifications to the TEI element classes.

Remarks:
This entity is embedded in the TEI DTDs before the TEI element classes are embedded. By default, the entity expands to the empty string; the user can override this default by declaring the entity with an appropriate value, typically this will take the form

```
<!ENTITY % TEI.extensions.ent system 'project.ent' >
```

DTD file: tei2
Referred to by:
Expansion: "
Discussed in: 3.6 ('The TEI2.DTD File') on p. 45

Entity Name: TEI.figures.dtd (i.e. TEI figures-base DTD) TEI.figures.dtd

Description: identifies the file containing element and attribute list declarations for the base tag set for tables, formulas, and graphics.

Remarks:
This entity is declared with this value when the user includes the base tag set for formulas and tables.

DTD file: tei2
Referred to by:
Expansion: system 'teiform2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

TEI.figures.ent

Entity Name: TEI.figures.ent (i.e. TEI figures-base entities)

Description: identifies the file containing parameter entity declarations for the element classes defined in the additional tag set for figures.

Remarks:
This entity is declared with this value when the user includes the additional tag set for figures, tables, and formulae.

DTD file: teiclas2.ent
Referred to by:
Expansion: system 'teifig2.ent'
Discussed in: 3.7 ('Element Classes') on p. 51

TEI.figures

Entity Name: TEI.figures (i.e. TEI figures and tables DTD fragment)

Description: controls the inclusion, in the DTD, of element and attribute declarations for figures and tables.

Remarks:
To include elements and attributes for figures and tables in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: teikey2.ent
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.5 ('Global Attributes') on p. 42; 3.6 ('The TEI2.DTD File') on p. 45

TEI.figures

Entity Name: TEI.figures (i.e. TEI graphics and figures DTD fragment)

Description: controls the inclusion, in the DTD, of element and attribute declarations for graphics and figures.

Remarks:
To include elements and attributes for graphics and figures in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: teikey2.ent
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.5 ('Global Attributes') on p. 42; 3.6 ('The TEI2.DTD File') on p. 45

TEI.front.dtd

Entity Name: TEI.front.dtd

Description: defines the file in which tags for front matter are defined.

Remarks:
This parameter entity is used in each base tag set to include the front-matter tags, which are common to all bases.

DTD file: teistr2
Referred to by:
Expansion: system 'teifron2.dtd'
Discussed in: 7.4 ('Front Matter') on p. 201

TEI.fs.dtd

Entity Name: TEI.fs.dtd (i.e. TEI fs-base DTD)

Description: identifies the file containing element and attribute list declarations for the base tag set for feature structures.

Remarks:
This entity is declared with this value when the user includes the base tag set for feature structures.

DTD file: `teifsd2`
Referred to by:
Expansion: system 'teifs2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

Entity Name: TEI.fs (i.e. TEI feature structures DTD fragment) TEI.fs

Description: controls the inclusion, in the DTD, of element and attribute declarations for feature structures.

Remarks:
To include elements and attributes for feature structures in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: `teikey2.ent`
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.6 ('The TEI2.DTD File') on p. 45

Entity Name: TEI.general (i.e. TEI general base DTD fragment) TEI.general

Description: controls the inclusion, in the DTD, of element and attribute declarations for the "general" combined base.

Remarks:
To include elements and attributes for the "general" base in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: `teikey2.ent`
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.6 ('The TEI2.DTD File') on p. 45

Entity Name: TEI.general.dtd (i.e. TEI general-base DTD) TEI.general.dtd

Description: identifies the file containing element and attribute list declarations for the base tag set for the "general" base tag set.

Remarks:
This entity is declared with this value when the user includes the "general" mixed-base tag set.

DTD file: `tei2`
Referred to by:
Expansion: system 'teigen2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

Entity Name: TEI.header.dtd (i.e. TEI header) TEI.header.dtd

Description: identifies the file containing element and attribute list declarations for the TEI header.

Remarks:
This entity is included in all TEI DTDs.

DTD file: `teifsd2`
Referred to by:
Expansion: system 'teihdr2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

Entity Name: TEI.keywords.ent TEI.keywords.ent

Description: file containing the parameter entity declarations for TEI keywords.

Remarks:

The keywords defined in this file define non-SGML data types (e.g. ISO dates) and control the selection of base and additional tag sets.

DTD file: teitsd2
Referred to by:
Expansion: system 'teikey2.ent'
Discussed in: 3.8.3 ('Parameter Entities for TEI Keywords') on p. 67

TEI.linking.dtd

Entity Name: TEI.linking.dtd (i.e. TEI segmentation and alignment elements)

Description: identifies the file containing element and attribute list declarations for the additional tag set for segmentation and alignment.

Remarks:
This entity is declared with this value when the user includes the additional tag set for segmentation and alignment.

DTD file: tei2
Referred to by:
Expansion: system 'teilink2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

TEI.linking.ent

Entity Name: TEI.linking.ent (i.e. TEI linking-tag-set entities)

Description: identifies the file containing parameter entity declarations for the element classes defined in the additional tag set for segmentation and alignment.

Remarks:
This entity is declared with this value when the user includes the additional tag set for segmentation and alignment.

DTD file: teiclas2.ent
Referred to by:
Expansion: system 'teilink2.ent'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

TEI.linking

Entity Name: TEI.linking (i.e. TEI DTD fragment for linking tag set)

Description: controls the inclusion, in the DTD, of element and attribute declarations for linking, segmentation and alignment.

Remarks:
To include elements and attributes for segmentation and alignment in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: teikey2.ent
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.6 ('The TEI2.DTD File') on p. 45

TEI.mixed

Entity Name: TEI.mixed (i.e. TEI 'mixed' base DTD fragment)

Description: controls the inclusion, in the DTD, of element and attribute declarations for the "mixed" combined base.

Remarks:
To include elements and attributes for the "mixed" base in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: teikey2.ent
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.6 ('The TEI2.DTD File') on p. 45

TEI.mixed.dtd

Entity Name: TEI.mixed.dtd (i.e. TEI mixed-base DTD)

Description: identifies the file containing element and attribute list declarations for the base tag set for the "mixed" base tag set.

Remarks:
This entity is declared with this value when the user includes the base tag set for the "mixed" base.

DTD file: tei2
Referred to by:
Expansion: system 'teimix2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

Entity Name: TEI.names.dates.dtd (i.e. TEI names.dates-base DTD) TEI.names.dates.dtd

Description: identifies the file containing element and attribute list declarations for the base tag set for detailed analysis of names and dates.

Remarks:
This entity is declared with this value when the user includes the base tag set for names and dates.

DTD file: tei2
Referred to by:
Expansion: system 'teind2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

Entity Name: TEI.names.dates.ent (i.e. parameter entities for names and dates) TEI.names.dates.ent

Description: identifies the file containing parameter entity declarations for the element classes defined in the additional tag set for names and dates.

Remarks:
This entity is declared with this value when the user includes the additional tag set for names and dates.

DTD file: teiclas2.ent
Referred to by:
Expansion: system 'teind2.ent'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

Entity Name: TEI.names.dates (i.e. TEI names and dates DTD fragment) TEI.names.dates

Description: controls the inclusion, in the DTD, of element and attribute declarations for detailed analysis of names and dates.

Remarks:
To include elements and attributes for names and dates in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: teikey2.ent
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.5 ('Global Attributes') on p. 42; 3.6 ('The TEI2.DTD File') on p. 45

Entity Name: TEI.nets.dtd (i.e. TEI networks-base DTD) TEI.nets.dtd

Description: identifies the file containing element and attribute list declarations for the base tag set for graph theory (graphs, digraphs and other networks).

Remarks:
This entity is declared with this value when the user includes the base tag set for graph theory.

DTD file: tei2
Referred to by:
Expansion: system 'teinet2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

Entity Name: TEI.nets (i.e. TEI graph theoretic DTD fragment) TEI.nets

Description: controls the inclusion, in the DTD, of element and attribute declarations for graph theory (graphs, digraphs and other networks)

Remarks:
To include elements and attributes for graphs and digraphs in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: teikey2.ent
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.5 ('Global Attributes') on p. 42; 3.6 ('The TEI2.DTD File') on p. 45

TEI.prose.dtd

Entity Name: TEI.prose.dtd (i.e. TEI prose-base DTD)
Description: identifies the file containing element and attribute list declarations for the base tag set for prose.

Remarks:
This entity is declared with this value when the user includes the base tag set for prose.

DTD file: tei2
Referred to by:
Expansion: system 'teipros2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

TEI.prose

Entity Name: TEI.prose (i.e. TEI prose DTD fragment)
Description: controls the inclusion, in the DTD, of element and attribute declarations for prose.

Remarks:
To include elements and attributes for prose in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: teikey2.ent
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36; 3.6 ('The TEI2.DTD File') on p. 45

TEI.singleBase

Entity Name: TEI.singleBase (i.e. single-base flag)
Description: controls the inclusion, in the DTD, of the default text structure elements. Its value is IGNORE when the mixed base or general base is selected.

Remarks:
The user should not redefine this parameter entity.

DTD file: teiclas2.ent
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.8 ('Other Parameter Entities in TEI DTDs') on p. 65

TEI.singleBase

Entity Name: TEI.singleBase (i.e. single-base flag)
Description: controls the inclusion, in the DTD, of the default text structure elements. Its value is INCLUDE when only a single base tag set is selected.

Remarks:
The user should not redefine this parameter entity.

DTD file: teiclas2.ent
Referred to by:
Expansion: 'INCLUDE'
Discussed in: 3.8 ('Other Parameter Entities in TEI DTDs') on p. 65

TEI.spoken.dtd

Entity Name: TEI.spoken.dtd (i.e. TEI spoken-base DTD)
Description: identifies the file containing element and attribute list declarations for the base tag set for spoken texts.

Remarks:

This entity is declared with this value when the user includes the base tag set for spoken texts.

DTD file: tei2
Referred to by:
Expansion: system 'teispok2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

Entity Name: TEI.spoken.ent (i.e. TEI spoken-base entities) TEI.spoken.ent

Description: identifies the file containing parameter entity declarations for the element classes defined in the base tag set for spoken texts.

Remarks:
This entity is declared with this value when the user includes the base tag set for spoken texts.

DTD file: teiclas2.ent
Referred to by:
Expansion: system 'teispok2.ent'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

Entity Name: TEI.spoken (i.e. TEI spoken texts DTD fragment) TEI.spoken

Description: controls the inclusion, in the DTD, of element and attribute declarations for spoken texts.

Remarks:
To include elements and attributes for spoken texts in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: teikey2.ent
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.5 ('Global Attributes') on p. 42; 3.6 ('The TEI2.DTD File') on p. 45

Entity Name: TEI.structure.dtd TEI.structure.dtd

Description: defines the file in which the default text structure used by many base tag sets is defined.

Remarks:

DTD file: teiterm2
Referred to by:
Expansion: system 'teistr2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

Entity Name: TEI.terminology (i.e. TEI terminological data DTD fragment) TEI.terminology

Description: controls the inclusion, in the DTD, of element and attribute declarations for terminological data.

Remarks:
To include elements and attributes for terminological data in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: teikey2.ent
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.6 ('The TEI2.DTD File') on p. 45

Entity Name: TEI.terminology.dtd (i.e. TEI terminology-base DTD) TEI.terminology.dtd

Description: identifies the file containing element and attribute list declarations for the base tag set for terminological data.

Remarks:
This entity is declared with this value when the user includes the base tag set for terminological data.

DTD file: tei2
Referred to by:
Expansion: system 'teiterm2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

TEI.terminology.ent

Entity Name: TEI.terminology.ent (i.e. TEI terminology-base entities)
Description: identifies the file containing parameter entity declarations for the element classes defined in the base tag set for terminological data.
Remarks:
This entity is declared with this value when the user includes the base tag set for terminological data.
DTD file: teiclas2.ent
Referred to by:
Expansion: system 'teiterm2.ent'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

TEI.textcrit.dtd

Entity Name: TEI.textcrit.dtd (i.e. TEI text criticism base DTD)
Description: identifies the file containing element and attribute list declarations for the additional tag set for text criticism.
Remarks:
This entity is declared with this value when the user includes the additional tag set for text criticism.
DTD file: tei2
Referred to by:
Expansion: system 'teitc2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

TEI.textcrit.ent

Entity Name: TEI.textcrit.ent (i.e. TEI text criticism entities)
Description: identifies the file containing parameter entity declarations for the element classes defined in the additional tag set for text criticism.
Remarks:
This entity is declared with this value when the user includes the additional tag set for text criticism.
DTD file: teiclas2.ent
Referred to by:
Expansion: system 'teitc2.ent'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

TEI.textcrit

Entity Name: TEI.textcrit (i.e. TEI text criticism DTD fragment)
Description: controls the inclusion, in the DTD, of element and attribute declarations for text criticism.
Remarks:
To include elements and attributes for text criticism in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.
DTD file: teikey2.ent
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.6 ('The TEI2.DTD File') on p. 45

TEI.transcr.dtd

Entity Name: TEI.transcr.dtd (i.e. TEI DTD for transcription of primary sources)
Description: identifies the file containing element and attribute list declarations for the base tag set for description of the source text.
Remarks:
This entity is declared with this value when the user includes the base tag set for transcription of primary sources

DTD file: tei2
Referred to by:
Expansion: system 'teitran2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

Entity Name: TEI.transcr.ent (i.e. TEI entities for transcription of primary sources) TEI.transcr.ent

Description: identifies the file containing parameter entity declarations for the element classes defined in the additional tag set for transcription of primary sources.

Remarks:
This entity is declared with this value when the user includes the additional tag set for transcription of primary sources.

DTD file: teiclas2.ent
Referred to by:
Expansion: system 'teitran2.ent'
Discussed in: 3.7 ('Element Classes') on p. 51

Entity Name: TEI.transcr (i.e. TEI DTD fragment for transcription of primary sources) TEI.transcr

Description: controls the inclusion, in the DTD, of element and attribute declarations for transcription of the source text.

Remarks:
To include elements and attributes for transcription in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: teikey2.ent
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.5 ('Global Attributes') on p. 42; 3.6 ('The TEI2.DTD File') on p. 45

Entity Name: TEI.verse (i.e. TEI verse DTD fragment) TEI.verse

Description: controls the inclusion, in the DTD, of element and attribute declarations for verse.

Remarks:
To include elements and attributes for verse in the DTD, the user should declare this entity with a value of "INCLUDE"; this will override the default.

DTD file: teikey2.ent
Referred to by:
Expansion: 'IGNORE'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36; 3.6 ('The TEI2.DTD File') on p. 45

Entity Name: TEI.verse.dtd (i.e. TEI verse-base DTD) TEI.verse.dtd

Description: identifies the file containing element and attribute list declarations for the base tag set for verse.

Remarks:
This entity is declared with this value when the user includes the base tag set for verse.

DTD file: tei2
Referred to by:
Expansion: system 'teivers2.dtd'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

Entity Name: TEI.verse.ent (i.e. TEI verse-base entities) TEI.verse.ent

Description: identifies the file containing parameter entity declarations for the element classes defined in the base tag set for verse.

Remarks:
This entity is declared with this value when the user includes the base tag set for verse.

DTD file: teiclas2.ent
Referred to by:
Expansion: system 'teivers2.ent'
Discussed in: 3.2 ('Core, Base, and Additional Tag Sets') on p. 36

TEI.wsdNames

Entity Name: TEI.wsdNames

Description: file containing parameter entity declarations for all generic identifiers used in the writing system declaration

Remarks:
The parameter entities in this file all take the same form as the two shown below:

```
<!ENTITY % n.div0 'div0' >
<!ENTITY % n.div1 'div1' >
```

Element and attribute-list declarations in the DTDs refer to the parameter entity *n.div1*, not directly to the generic identifier *div1*. As a result, the declarations will function as desired even if a new generic identifier is substituted (e.g. "caput" for "div1" and "liber" for "div0":

```
<!ENTITY % n.div0 'liber' >
<!ENTITY % n.div1 'caput' >
```

This allows generic identifiers to be renamed conveniently, to provide names in languages other than English, or to provide shorter names than those documented here. See further chapter 29 ('Modifying the TEI DTD') on p. 619.

DTD file: teiwsd2
Referred to by:
Expansion: system 'wdgis2.ent'
Discussed in: 25.1 ('Overall Structure of Writing System Declaration') on p. 571

termtags

Entity Name: termtags

Description: system entity with definitions for basic terminology tags.

Remarks:
The default definition of this entity is used to invoke the element declarations for the "nested" style of terminological markup. To invoke the alternative "flat" style declarations, this entity should be defined as "system 'teite2f.dtd"'.

DTD file: teiterm2
Referred to by:
Expansion: system 'teite2n.dtd'
Discussed in: 13.4 ('Overall Structure of Terminological Documents') on p. 319

version

Entity Name: version (i.e. version)

Description: defines the name to be used for the root element of a concurrent markup stream for marking pages and lines of some reference edition.

Remarks:
Some name for the edition should be supplied by defining this parameter entity within the appropriate DTD subset. If none is defined, the parameter entity (and thus the document type) default to "ref".

DTD file: teipl2
Referred to by:
Expansion: 'ref'
Discussed in: 31.6 ('Concurrent Markup for Pages and Lines') on p. 637

Chapter 35

Elements

<abbr> (i.e. abbreviation) contains an abbreviation of any sort. abbr

Attributes:

expan (i.e. expansion) gives an expansion of the abbreviation.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED
Example:

```
The address is Southmoor <abbr expan='road'>Rd</>.
```

Only one expansion may be given for an abbreviation; if different expansions are to be proposed, the tags for critical apparatus should be used.

resp (i.e. responsibility) signifies the editor or transcriber responsible for supplying the expansion of the abbreviation held as the value of the expan attribute.
Data type: IDREF
Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).
Default: %INHERITED
Example:

```
The address is Southmoor <abbr expan='Road' resp=LB>Rd</>.
```

If no expansion is given, the resp attribute has no meaning.

cert (i.e. certainty) signifies the degree of certainty ascribed to the expansion of the abbreviation.
Data type: CDATA
Default: #IMPLIED

type allows the encoder to classify the abbreviation according to some convenient typology.
Data type: CDATA
Sample values include:

suspension the abbreviation provides the first letter(s) of the word or phrase, omitting the remainder.
contraction the abbreviation omits some letter(s) in the middle.
brevigraph the abbreviation comprises a special symbol or mark.
superscription the abbreviation includes writing above the line.
acronym the abbreviation comprises the initial letters of the words of a phrase.
title the abbreviation is for a title of address (Dr, Ms, Mr, ...)
organization the abbreviation is for the name of an organization.
geographic the abbreviation is for a geographic name.

Default: #IMPLIED
Example:

```
<abbr type=acronym>RSVP</abbr>
<abbr type=organization>SPQR</abbr>
<abbr type=brevigraph>&per;</abbr>
<abbr type=contraction>yr hbl servt</abbr>
```

The type attribute is provided for the sake of those who wish to classify abbreviations at their point of occurrence; this may be useful in some circumstances, though usually the same abbreviation will have the same type in all occurrences. As the sample values make clear, abbreviations may be classified by the method used to construct them, the method of writing them, or the referent of the term abbreviated; the typology used is up to the encoder and should be carefully planned to meet the needs of the expected use.

Example:

```
<abbr type=organization
  expan='senatus populusque romanorum'
  resp=author>SPQR</abbr>
```

This tag is the mirror image of the <expan> tag; both allow the encoder to transcribe both an abbreviation and its expansion. In <abbr>, however, the original is transcribed as the content of the element and the expansion as an attribute value; <expan> reverses this. The choice between the two is up to the user.

The <abbr> tag is not required; if appropriate, the encoder may transcribe abbreviations in the source text silently, without tagging them. If abbreviations are not transcribed directly but *expanded* silently, then the TEI header should so indicate.

For a typology of ME abbreviations, see A. G. Petty, *English literary hands from Chaucer to Dryden* (London: Edward Arnold, 1977), pp. 22-25.

Part: additional tag set for common core features

Member of classes: data [and indirectly also:] phrase

DTD file: teicore2

Data description: May contain character data and phrase-level elements.

Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT abbr          - -  (%phrase.seq;)                    >

<!ATTLIST abbr               %a.global
          expan              CDATA                 #IMPLIED
          resp               IDREF                 %INHERITED

          cert               CDATA                 #IMPLIED
          type               CDATA                 #IMPLIED    >
```

Discussed in: 6.4.5 ('Abbreviations and Their Expansions') on p. 139

activity

<activity> (i.e. activity) contains a brief informal description of what a participant in a language interaction is doing other than speaking, if anything.

Attributes: [None: global and inherited attributes only.]

Example:

```
<activity>driving</activity>
```

For more fine-grained description of participant activities during a spoken text, the <event> element should be used.
Part: auxiliary tag set for corpora and collections
Member of classes:
DTD file: teicorp2
Data description: May contain character data and phrase-level elements.
Occurs within: setting
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT activity      - O  (%phrase.seq)                  >
<!ATTLIST activity           %a.global                      >
```

Discussed in: 23.2.3 ('The Setting Description') on p. 549

<actor> (i.e. actor) Name of an actor appearing within a cast list. — actor

Attributes: [None: global and inherited attributes only.]
Example:

```
<castItem>
 <role>Mathias</role><roleDesc>the Burgomaster</roledesc>
 <actor who=IRVH1>Mr. Henry Irving</actor>
</castItem>
```

Part: base tag set for performance texts
Member of classes:
DTD file: teidram2
Data description: Contains phrase level elements and character data only.
Occurs within: castItem
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT actor         - O  (%phrase.seq)                  >
<!ATTLIST actor              %a.global                      >
```

Discussed in: 10.1.4 ('Cast Lists') on p. 233

<add> (i.e. addition) contains letters, words, or phrases inserted in the text by an author, scribe, annotator or corrector. — add

Attributes:

place if the the addition is written into the copy text, indicates where the additional text is written.
Data type: CDATA
Sample values include:

inline addition is made in a space left in the witness by an earlier scribe
supralinear addition is made above the line
infralinear addition is made below the line
left addition is made in left margin
right addition is made in right margin
top addition is made in top margin
bottom addition is made in bottom margin
opposite addition is made on opposite page
verso addition is made on verso of sheet

mixed addition is made somewhere, one or more of other values

Default: #IMPLIED

resp (i.e. responsible) signifies the editor or transcriber responsible for identifying the hand of the addition.

Data type: IDREF

Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).

Default: %INHERITED

cert (i.e. certainty) signifies the degree of certainty ascribed to the identification of the hand of the addition.

Data type: CDATA

Default: #IMPLIED

hand signifies the hand of the agent which made the addition.

Data type: IDREF

Value: must be one of the hand identifiers declared in the document header (see section 18.2.1 ('Document Hands') on p. 456).

Default: %INHERITED

The <add> tag should not be used for additions made by editors or encoders. In these cases, either the <corr> tag or the <supplied> tag should be used.

Part: additional tag set for common core features

Member of classes: edit [and indirectly also:] phrase

DTD file: teicore2

Data description: May contain character data and phrase-level elements.

Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph entry entryFree event expan eTree figure foreign formula gap gi gloss graph handShift hi kinesic l label lang lg lg1 link list listBibl m measure mentioned move name note num orig oRef oVar p pause phr ptr pRef pVar q quote ref reg rs s seg shift sic sound soCalled sp stage superentry table tag tech term termEntry text time timeRange timeStruct title tree u val view vocal w writing xptr xref TEI...end

Declaration:

```
<!ELEMENT add           - -  (%specialPara;)                    >

<!ATTLIST add                 %a.global
          place               CDATA                 #IMPLIED
          resp                IDREF                 %INHERITED

          cert                CDATA                 #IMPLIED
          hand                IDREF                 %INHERITED    >
```

Discussed in: 6.5.3 ('Additions, Deletions and Omissions') on p. 144

addName

<addName> (i.e. additional name) contains an additional name component, such as a nickname, epithet, or alias, or any other descriptive phrase used within a personal name.

Attributes: [None: global and inherited attributes only.]

Example:

```
<persName>
  <foreName>Frederick</foreName>
  <addName type=epithet>the Great</persName>
  <roleName>Emperor of Prussia</roleName>
</persName>
```

Part: additional tag set for names and dates
Member of classes: personPart [and indirectly also:] names
DTD file: teind2
Occurs within: persName
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT addName       - -  (%phrase.seq;)                >
<!ATTLIST addName            %a.global
                             %a.personPart                 >
```

Discussed in: 20.1 ('Personal Names') on p. 488

address

<address> contains a postal or other address, for example of a publisher, an organization, or an individual.

Attributes: [None: global and inherited attributes only.]
Example:

```
<address><street>via Marsala 24</>
        <postcode>40126</>
        <name>Bologna</>
        <country n=I>Italy</>
</address>
```

Example:

```
<address>
<addrLine>Computing Center, MC 135
<addrLine>P.O. Box 6998</>
<addrLine>Chicago, IL</>
<addrLine>60680 USA</>
</address>
```

Addresses may be encoded either as a sequence of lines, or using any sequence of address component elements.
Part: base tag set for common core features
Member of classes: data [and indirectly also:] phrase
DTD file: teicore2
Data description: If given as running prose, use a consistent format wherever possible, for example separating lines of the address by commas, and including any postal code in the standard form.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateline dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publicationStmt publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: addrLine postBox postCode street
Declaration:

```
<!ELEMENT address       - O  (addrLine+ | (%m.addrPart)*)       >
<!ATTLIST address             %a.global                         >
```

Discussed in: 6.4.2 ('Addresses') on p. 134; 5.2.4 ('Publication, Distribution, etc') on p. 86; 6.10.2.3 ('Imprint, Pagination, and Other Details') on p. 171

addrLine

<addrLine> contains one line of a postal or other address.

Attributes: [None: global and inherited attributes only.]
Example:

```
<address>
<addrLine>Computing Center, MC 135
<addrLine>P.O. Box 6998</>
<addrLine>Chicago, IL</>
<addrLine>60680 USA</>
</address>
```

Addresses may be encoded either as a sequence of lines, or using any sequence of address component elements.
Part: base tag set for common core features
Member of classes:
DTD file: teicore2
Data description:
Occurs within: address
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT addrLine      - O  (%phrase.seq)                      >
<!ATTLIST addrLine            %a.global                         >
```

Discussed in: 6.4.2 ('Addresses') on p. 134; 5.2.4 ('Publication, Distribution, etc') on p. 86; 6.10.2.3 ('Imprint, Pagination, and Other Details') on p. 171

addSpan

<addSpan> (i.e. added span of text) marks the beginning of a longer sequence of text added by an author, scribe, annotator or corrector (see also <add>).

Attributes:

type classifies the addition, using any convenient typology.
Data type: CDATA
Value: Any string of characters.
Default: #IMPLIED

place indicates where the addition is made.
Data type: CDATA
Sample values include:

inline addition is made in a space left in the witness by an earlier scribe.
supralinear addition is made above the line.
infralinear addition is made below the line.
marginleft addition is made in left margin.
marginright addition is made in right margin.
margintop addition is made in top margin.
marginbot addition is made in bottom margin.
overleaf addition is made on the other side of the leaf.

Default: #IMPLIED
Example:

```
Then they went back home.
<addSpan place='supralinear marginright overleaf' to=p23>
When they got there, ...
```

Long additions may frequently spill from one location to another; in this case, more than one value may be given, as shown in the example above.

resp (i.e. responsible) signifies the editor or transcriber responsible for identifying the hand of the addition.
Data type: IDREF
Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).
Default: %INHERITED

cert (i.e. certainty) signifies the degree of certainty ascribed to the identification of the hand of the addition.
Data type: CDATA
Default: #IMPLIED

hand signifies the hand of the agent which made the addition.
Data type: IDREF
Value: must be one of the hand identifiers declared in the document header (see section 18.2.1 ('Document Hands') on p. 456).
Default: %INHERITED

to identifies the endpoint of the added passage, by giving the ID of an <anchor> or other empty element placed there.
Data type: IDREF
Value: any valid SGML identifier, used on a later element.
Default: #REQUIRED
Both the beginning and the end of the added material must be marked; the beginning by the <addSpan> element itself, the end by the to attribute.

(optional)

Part: additional tag set for text criticism
Member of classes:
DTD file: teitran2
Data description: Empty.
Occurs within: [none]
May contain: [none]
Declaration:

```
<!ELEMENT addSpan       - O  EMPTY                          >

<!ATTLIST addSpan            %a.global
          type               CDATA              #IMPLIED
          place              CDATA              #IMPLIED
          resp               IDREF              %INHERITED

          cert               CDATA              #IMPLIED
          hand               IDREF              %INHERITED

          to                 IDREF              #REQUIRED      >
```

Discussed in: 18.1.4 ('Additions and Deletions') on p. 449

<admin> (i.e. administrative information) within a <termEntry> element, contains administrative information pertaining to data management and documentation of the entry.

Attributes:

type identifies the administrative event or information using some typology, preferably the dictionary of data element types specified in ISO WD 12 620.
Data type: CDATA
Sample values include:

responsibility The <admin> element identifies the agency or individual responsible for the data element or entry.

created The <admin> element describes the creation of the data element or entry.

updated The <admin> element describes the update or modification of the data element or entry.

approved The <admin> element describes the final approval of the data element or entry.

domain The element indicates the subject area to which a concept pertains.

subdomain The element indicates the subdomain of the subject area to which the concept pertains.

Default: #IMPLIED

A much fuller list of values for the type attribute may be generated from the dictionary of data element types under preparation as ISO TC 37/SC 3/WD 12 620, Computational Aids in Terminology. See ISO 12 620 for fuller details.

date indicates the date of the administrative event or information marked by the element.

Data type: CDATA

Value: a date in ISO standard form (yyyy-mm-dd).

Default: %ISO-DATE

The date attribute should be used on the <admin> element to indicate the date of the administrative process (creation, update, approval, etc.) being recorded, rather than a separate <date> element linked to the <admin> element by a depend attribute.

resp (i.e. responsibility) indicates the agency or individual responsible for the entry or data element on which the <admin> element depends, or for the administrative procedure recorded by the <admin> element.

Data type: CDATA

Value: any string of characters (usually the acronym of the agency or the initials of an individual).

Default: #IMPLIED

Some terminological database systems treat responsibility as a cross reference to a personnel entry.

For domain or other similar information, the <admin> element should be used to provide the domain information as the content of the element:

Example:

```
<admin depend=te84.11
       type='domain'
       resp='ISO/TC 61, Plastics'>plastics</admin>
```

In the case of responsibility information or information about maintenance of the entry, the <admin> element may have no content at all, its information all having been conveyed by the attributes type, date, and resp:

Example:

```
<admin depend=te84.11
       type='responsibility'
       resp='ISO/TC 61, Plastics'></admin>
```

Example:

```
<admin depend=te84.11
       type='created'
       date='1991-10-23'
       resp='SEW'></admin>
<admin depend=te84.11
       type='updated'
       date='1992-12-15'
       resp='MSM'></admin>
```

Administrative data take widely varying forms in different terminological databases; the type attribute should be used to indicate the particular class of administrative information involved. The <admin> element should also be used to record dates of major changes to the term entry, if these are to be recorded.

Part: base tag set for terminological data

Member of classes: terminologyMisc

DTD file: teite2n teite2f
Data description: May contain character data and phrase-level elements.
Occurs within: ofig termEntry tig
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT admin          - O  (%paraContent;)                  >

<!ATTLIST admin               %a.global
          type                CDATA               #IMPLIED
          date                CDATA               %ISO-date

          resp                CDATA               #IMPLIED      >
```

Discussed in: 13.4.2 ('DTD Fragment for Flat Style') on p. 322; 13.4.1 ('DTD Fragment for Nested Style') on p. 321; 13.2 ('Tags for Terminological Data') on p. 312

<affiliation> (i.e. affiliation) contains an informal description of a person's present or past affiliation with some organization, for example an employer or sponsor. affiliation

Attributes: [None: global and inherited attributes only.]
Example:

```
<affiliation>Junior project officer for the US
 <org>National Endowment for the Humanities</org>
</affiliation>
```

If included, the name of the organization should be tagged using the <org> element.
Part: auxiliary tag set for corpora and collections
Member of classes:
DTD file: teicorp2
Data description: May contain character data and phrase-level elements.
Occurs within: [none]
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT affiliation    - -  (%phrase.seq)                    >

<!ATTLIST affiliation         %a.global                        >
```

Discussed in: 23.2.2 ('The Participants Description') on p. 545

<alt> (i.e. alternation) identifies an alternation or a set of choices among elements or passages. alt

Attributes:

targets specifies the SGML identifiers of the alternative elements or passages.
Data type: IDREFS
Value: Each value specified must be the same as that specified as value for an ID attribute for some other element in the current SGML document.
Default: #REQUIRED

mode states whether the alternations gathered in this collection are exclusive or inclusive.
Data type: (EXCL | INCL)
Sample values include:

excl indicates that the alternation is exclusive, i.e. that at most one of the alternatives occurs.

incl indicates that the alternation is not exclusive, i.e. that one or more of the alternatives occur.

Default: %INHERITED

weights If mode=excl, each weight states the probability that the corresponding alternative occurs. If mode=incl each weight states the probability that the corresponding alternative occurs given that at least one of the other alternatives occurs.
Data type: NUMBERS
Value: a list of numbers, in the range from 0 to 1 if wScale=real, and in the range from 0 to 100 if wScale=perc.
Default: #IMPLIED
If mode=excl then sum of weights must be 1 (or 100%). If mode=incl, then sum of weights must be in the range from 0 to the number of alternants (or 0 to 100% times the number of alternants).

wScale indicates the scale used to express the value of the weights attribute value.
Data type: (PERC | REAL)
Sample values include:

perc indicates that the weights are expressed as percentages.

real indicates that the weights are expressed as values between 0 and 1.

Default: %INHERITED

targOrder specifies whether the targets are ordered.
Data type: (Y | N)
Sample values include:

Y indicates that the targets are ordered.

N indicates that the targets are not ordered.

Default: N
This attribute is one of those assigned to elements of the pointer class. It is specified here so as to assign it a default value.

Example:

```
<alt excl=Y targType='u u' targets='we.fun we.sun' weights='50 50'>
```

Part: additional tag set for
Member of classes: complexVal, pointer [and indirectly also:] featureVal
DTD file: teilink2.dtd
Data description: empty
Occurs within: altGrp f fs if vDefault vRange
May contain: [none]
Declaration:

```
<!ELEMENT alt           - O  EMPTY                              >

<!ATTLIST alt                %a.global
          type               CDATA                #IMPLIED
          resp               CDATA                #IMPLIED
          crdate             CDATA                #IMPLIED
          targType           NAMES                #IMPLIED
          evaluate           (all | one | none)   #IMPLIED
          targets            IDREFS               #REQUIRED
          mode               (excl | incl)        %INHERITED

          weights            NUMBERS              #IMPLIED
          wScale             (perc | real)        %INHERITED

          targOrder          (Y | N)              N              >
```

Discussed in: 14.8 ('Alternation') on p. 373

altGrp

<altGrp> (i.e. alternation group) groups a collection of <alt> elements and possibly pointers.

Attributes:

mode states whether the alternations gathered in this collection are exclusive or inclusive.
Data type: (EXCL | INCL)
Sample values include:

excl indicates that the alternation is exclusive, i.e. that at most one of the alternatives occurs.

incl indicates that the alternation is not exclusive, i.e. that one or more of the alternatives occur.

Default: EXCL

wScale indicates the scale used to express the value of the weights attribute value.

Data type: (PERC | REAL)

Sample values include:

perc indicates that the weights are expressed as percentages.

real indicates that the weights are expressed as values between 0 and 1.

Default: PERC

Example:

```
<altGrp mode=excl>
  <alt targType='seg seg' targets='dm lt bb' weights='50 25 25'>
  <alt targType='l l' targets='rl db' weights='50 50'>
</altGrp>
```

Example:

```
<altGrp mode=incl wscale=perc targType='seg l'>
  <alt targets='dm rl' weights='90 90'>
  <alt targets='lt rl' weights='5 5'>
  <alt targets='bb rl' weights='5 5'>
  <alt targets='dm db' weights='10 10'>
  <alt targets='lt db' weights='45 90'>
  <alt targets='bb db' weights='45 90'>
</altGrp>
```

Part: additional tag set for
Member of classes: pointerGroup [and indirectly also:] pointer
DTD file: teilink2.dtd
Data description: Any number of alternations, pointers or extended pointers.
Occurs within: [none]
May contain: alt ptr xptr
Declaration:

```
<!ELEMENT altGrp        - -  ((alt | ptr | xptr)*)              >
<!ATTLIST altGrp             %a.global
                             %a.pointerGroup
          mode               (excl | incl)       excl
          wScale             (perc | real)       perc           >
```

Discussed in: 14.8 ('Alternation') on p. 373

analytic

<analytic> (i.e. analytic level) contains bibliographic elements describing an item (e.g. an article or poem) published within a monograph or journal and not as an independent publication.

Attributes: [None: global and inherited attributes only.]

The <analytic> element may occur only within bibliographic citation or reference elements; it is mandatory for description of the analytic level of <bibl.struct> elements.

Part: base tag set for common core features
Member of classes: biblPart
DTD file: teicore2
Data description: May contain titles and statements of responsibility (author, editor, or other), in any order.
Occurs within: bibl biblStruct
May contain: author editor respStmt title
Declaration:

```
<!ELEMENT analytic      - O  (author | editor | respStmt |
                              title)*                          >
```

```
<!ATTLIST analytic          %a.global                       >
```

Discussed in: 6.10.2.1 ('Analytic, Monographic, and Series Levels') on p. 166

anchor

<anchor> (i.e. anchor point) attaches an identifier to a point within a text, whether or not it corresponds with a textual element.

Attributes:

id (i.e. identifier) supplies an arbitrary name, unique within the document, used to identify the point at which this element occurs.
Data type: ID
Value: any valid name
Default: #REQUIRED
IDs may be chosen freely; note in particular that if they include numeric values there is no requirement for these to be in sequence.

Example:

```
<s>The anchor is he<anchor id=A234>re somewhere.</s>
<s>Help me find it.<ptr target=A234></s>
```

Part: additional tag set for segmentation, alignment, and linking
Member of classes: seg [and indirectly also:] phrase
DTD file: dummy teilink2.dtd
Data description: empty
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref [May appear anywhere within: entry]
May contain: [none]
Declaration:

```
<!ELEMENT anchor        - O  EMPTY                           >

<!ATTLIST anchor             %a.analysis
                             %a.linking
                             %a.terminology
          n                  CDATA              #IMPLIED
          lang               IDREF              %INHERITED

          rend               CDATA              #IMPLIED
                             %a.seg
          id                 ID                 #REQUIRED     >
```

Discussed in: 11.3.2 ('Synchronization and Overlap') on p. 262; 14.4 ('Correspondence and Alignment') on p. 358

any

<any> (i.e. Any value) represents boolean *true* value variable.

Attributes: [None: global and inherited attributes only.]
Example:

```
<f name=gender><any>
```

Part: additional tag set for feature structures
Member of classes: boolean [and indirectly also:] singleVal
DTD file: teifs2

Data description: Empty element.
Occurs within: f fvLib vAlt
May contain: [none]
Declaration:

```
<!ELEMENT any            - O  EMPTY                          >

<!ATTLIST any                 %a.global                      >
```

Discussed in: 16.8 ('Boolean, Default and Uncertain Values') on p. 417

<app> (i.e. apparatus entry) contains one entry in a critical apparatus, with an optional lemma and at least one reading. app

Attributes:

type classifies the variation contained in this element according to some convenient typology.
Data type: CDATA
Value: Any convenient descriptive word or phrase, describing the extent of the variation (e.g. "word", "phrase", "punctuation", etc.) its text-critical significance (e.g. "significant", "accidental", "unclear"), or the nature of the variation or the principles required to understand it (e.g. "lectio difficilior", "usus auctoris", etc.)
Default: #IMPLIED

from identifies the beginning of the lemma in the base text, if necessary.
Data type: IDREF
Value: a valid SGML ID.
Default: #IMPLIED
This attribute is only used when the double-end point method of apparatus markup is used.

to identifies the endpoint of the lemma in the base text, if necessary.
Data type: IDREF
Value: a valid SGML ID.
Default: #IMPLIED
This attribute is only used when the double-end point method of apparatus markup is used, with the encoded apparatus held in a separate file rather than being embedded *in-line* in the base-text file.

loc (i.e. location) indicates the location of the variation, when the location-referenced method of apparatus markup is used.
Data type: CDATA
Value: Any string containing a canonical reference for the passage to which the variation applies.
Default: #IMPLIED
This attribute is used only when the location-referenced encoding method is used.

Part: additional tag set for text criticism
Member of classes:
DTD file: teitc2
Data description: May contain an optional lemma and one or more readings or reading groups.
Occurs within: [none]
May contain: lem rdg rdgGrp wit
Declaration:

```
<!ELEMENT app            - O  (lem?, ((rdg, wit?) | (rdgGrp,
                              wit?))+)                       >

<!ATTLIST app                 %a.global
          type                CDATA               #IMPLIED
          from                IDREF               #IMPLIED
          to                  IDREF               #IMPLIED
          loc                 CDATA               #IMPLIED   >
```

Discussed in: 19.1.1 ('The Apparatus Entry') on p. 469

arc

<arc> encodes an arc, the connection from one node to another in a graph.

Attributes:

label gives a label for an arc.
Data type: CDATA
Value: A character string.
Default: #IMPLIED

label2 gives a second label for an arc.
Data type: CDATA
Value: A character string.
Default: #IMPLIED
Use this attribute together with the label attribute if a transducer is being encoded.

from gives the ID of the node which is adjacent from this arc.
Data type: IDREF
Value: The ID of a node.
Default: #REQUIRED

to gives the ID of the node which is adjacent to this arc.
Data type: IDREF
Value: The ID of a node.
Default: #REQUIRED

Example:

```
<arc from=t3 to=t3 label='OLD' label2='VIEUX'>
```

The <arc> element must be used if the arcs are labeled. Otherwise, arcs can be encoded using the adj, adjTo and adjFrom attributes on the <node> tags in the graph. Both <arc> tags and adjacency attributes can be used, but the resulting encoding would be highly redundant.

Part: additional tag set for graphs, networks, and trees
Member of classes:
DTD file: teinet2
Data description: empty
Occurs within: graph
May contain: [none]
Declaration:

```
<!ELEMENT arc           - O  EMPTY                              >

<!ATTLIST arc                %a.global
          label              CDATA                #IMPLIED
          label2             CDATA                #IMPLIED
          from               IDREF                #REQUIRED
          to                 IDREF                #REQUIRED      >
```

Discussed in: 21.1 ('Graphs and Digraphs') on p. 506

argument

<argument> A formal list or prose description of the topics addressed by a subdivision of a text.

Attributes: [None: global and inherited attributes only.]
Example:

```
<argument>
<p>Monte Video &dash; Maldonado &dash; Excursion
to R Polanco &dash; Lazo and Bolas &dash; Partridges &dash;
Absence of Trees &dash; Deer &dash; Capybara, or River Hog &dash;
Tucutuco &dash; Molothrus, cuckoo-like habits &dash; Tyrant
Flycatcher &dash; Mocking-bird &dash; Carrion Hawks &dash;
Tubes formed by Lightning &dash; House struck
</argument>
```

Part: base tag set for common core features
Member of classes: divtop

DTD file: teistr2
Data description: Often contains either a list or a paragraph
Occurs within: back body castList div div0 div1 div2 div3 div4 div5 div6 div7 epilogue front group lg performance prologue
May contain: bibl biblFull biblStruct camera caption castList cit entry entryFree event eTree graph head kinesic l label lg lg1 list listBibl move note p pause q quote shift sound sp stage superentry tech termEntry tree u view vocal writing TEI...end
Declaration:

```
<!ELEMENT argument      - -  (head?, %component.seq;)          >
<!ATTLIST argument           %a.global                         >
```

Discussed in: 7.2.4 ('Content of Textual Divisions') on p. 194; 7.2 ('Elements Common to All Divisions') on p. 190

<att> contains the name of an attribute appearing within running text. — att

Attributes:

TEI indicates whether this attribute is part of the TEI scheme (i.e., defined in a TEI DTD fragment) or not.
Data type: (YES | NO)
Sample values include:

yes this attribute is part of the TEI scheme.
no this attribute is not part of the TEI scheme.

Default: YES

Example:

```
The TEI defines three <soCalled>global</soCalled> attributes;
their names are <att>id</att>, <att>rend</att> and <att>n</att>.
<att TEI=NO>type</att> is not among them.
```

Part: auxiliary tag set for tag set declarations
Member of classes: sgmlKeywords [and indirectly also:] phrase
DTD file: teitsd2
Data description: May contain a valid SGML name only.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA
Declaration:

```
<!ELEMENT att           - -  (#PCDATA)                         >
<!ATTLIST att                %a.global
          TEI                (yes | no)          yes           >
```

Discussed in: 27 ('Tag Set Documentation') on p. 601

<attDef> (i.e. attribute definition) contains the definition of a single attribute. — attDef

Attributes:

usage specifies the optionality of an attribute or element.
Data type: (REQ | MWA | REC | RWA | OPT)
Sample values include:

req required
mwa mandatory when applicable
rec recommended
rwa recommended when applicable
opt optional

Default: OPT

Example:

```
 <attDef usage=rec>
<attName>type</attName>
<desc>specifies a name conventionally used
for this level of subdivision, e.g. <q>act</q>,
<q>volume</q>, <q>book</q>, <q>section</q>, <q>canto</q>, etc.</desc>
<dataType>CDATA
<valDesc>any string of characters</valDesc>
<default>#CURRENT
</attDef>
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description:
Occurs within: attList
May contain: attName dataType default desc eg equiv remarks rs valDesc valList
Declaration:

```
<!ELEMENT attDef        - O  (attName, rs?, desc, (dataType,
                             (valList | valDesc)?), default,
                             eg?, remarks?, equiv*)            >

<!ATTLIST attDef             %a.global
          usage              (req | mwa | rec | rwa | opt)
                                                  opt          >
```

Discussed in: 27.1.1 ('The AttList Documentation Element') on p. 605

attlDecl

<attlDecl> (i.e. attList declaration) contains the SGML ATTLIST declaration associated with this element.

Attributes: [None: global and inherited attributes only.]

If the opening delimiter is parsed by the SGML parser, an error will result; it should therefore be represented by an entity reference, thus:

Example:

```
<attlDecl>
 &lt;!ATTLIST blort
              id       ID       #REQUIRED
              rend     CDATA    #IMPLIED   >
</attlDecl>
```

Alternatively, the entire element content may be enclosed in a CDATA marked section:

Example:

```
<attlDecl>
 <![ CDATA [
 <!ATTLIST blort
              id       ID       #REQUIRED
              rend     CDATA    #IMPLIED   >
 ]]>
</attlDecl>
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2

Data description: contains the full text of an SGML ATTLIST declaration; to avoid parsing errors, the opening delimiter must be given as an entity reference, or else the entire content of the element should be enclosed in a CDATA marked section.
Occurs within: tagDoc
May contain: #PCDATA
Declaration:

```
<!ELEMENT attlDecl      - -  (#PCDATA)                  >
<!ATTLIST attlDecl           %a.global                  >
```

Discussed in: 27.1 ('The TagDoc Documentation Element') on p. 603

<attList> contains documentation for all the attributes associated with this element, as a series of <attDef> elements. attList

Attributes: [None: global and inherited attributes only.]
Example:

```
<attList>
   <attDef> ... </attDef>
   <attDef> ... </attDef>
</attList>
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: contains a series of <attDef> elements only
Occurs within: classDoc tagDoc
May contain: attDef
Declaration:

```
<!ELEMENT attList       - O  (attDef*)                  >
<!ATTLIST attList            %a.global                  >
```

Discussed in: 27.1 ('The TagDoc Documentation Element') on p. 603; 27.2 ('Element Classes') on p. 606

<attName> (i.e. attribute name) contains the name of the attribute being defined by an <attDef> element. attName

Attributes: [None: global and inherited attributes only.]
Example:

```
<attName>type</attName>
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: must be a valid SGML name.
Occurs within: attDef
May contain: #PCDATA
Declaration:

```
<!ELEMENT attName       - O  (#PCDATA)                  >
<!ATTLIST attName            %a.global                  >
```

Discussed in: 27.1.1 ('The AttList Documentation Element') on p. 605

<author> in a bibliographic reference, contains the name of the author(s), personal or corporate, of a work; the primary *statement of responsibility* for any bibliographic item. author

Attributes: [None: global and inherited attributes only.]
Example:

```
<author>British Broadcasting Corporation
<author>La Fayette, Marie Madeleine Pioche de la Vergne,
        comtesse de (1634-1693)
```

Particularly where cataloguing is likely to be based on the content of the header, it is advisable to use generally recognized authority lists for the exact form of personal names.

In the case of a broadcast, use this element for the name of the company or network which broadcasts the program.

Part: base tag set for common core features
Member of classes: biblPart
DTD file: teicore2
Data description: Any string of characters and phrase-level tags.
Occurs within: analytic bibl monogr titleStmt
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT author        - O  (%phrase.seq)                    >

<!ATTLIST author             %a.global                        >
```

Discussed in: 6.10.2.2 ('Authors, Titles, and Editors') on p. 168; 5.2.1 ('The Title Statement') on p. 82

authority

<authority> (i.e. release authority) supplies the name of a person or other agency responsible for making an electronic file available, other than a publisher or distributor.

Attributes: [None: global and inherited attributes only.]
Example:

```
<authority>John Smith
```

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:
Occurs within: publicationStmt
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT authority     - O  (%phrase.seq;)                   >

<!ATTLIST authority          %a.global                        >
```

Discussed in: 5.2.4 ('Publication, Distribution, etc') on p. 86

availability

<availability> supplies information about the availability of a text, for example any restrictions on its use or distribution, its copyright status, etc.

Attributes:

status supplies a code identifying the current availability of the text.
Data type: (FREE | UNKNOWN | RESTRICTED)
Sample values include:

free the text is freely available.
unknown the status of the text is unknown.
restricted the text is not freely available.

Default: #IMPLIED

Example:

```
<availability status=restricted>
    <p>Available for academic research purposes only.
<availability status=free>
<availability status=restricted>
    <p>Available under licence from the publishers.
```

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description: A consistent format should be adopted
Occurs within: publicationStmt
May contain: p
Declaration:

```
<!ELEMENT availability  - O  (p+)                              >

<!ATTLIST availability       %a.global
          status             (free | unknown | restricted)
                                                  #IMPLIED     >
```

Discussed in: 5.2.4 ('Publication, Distribution, etc') on p. 86

<back> (i.e. back matter) contains any appendixes, etc. following the main part of a text. back

Attributes: [None: global and inherited attributes only.]
Example:

```
<back>
<div1 type='appendix'>
<head>>The Golden Dream or, the Ingenuous Confession
<p>To shew the Depravity of human Nature..
<!-- ... -->
</div1><div1 type=epistle>
<head>A letter from the Printer, which he desires
may be inserted
<salute>Sir.</salute>
<p>I have done with your Copy, so you may return
it to the Vatican, if you please
<!-- ... -->
<div1 type=advert>
<head>The Books usually read by the Scholars
of Mrs Two-Shoes are these and are sold at Mr
Newbery's at the Bible and Sun in St Paul's
Church-yard.</head>
<list>
<item n=1>The Christmas Box, Price 1d.
<item n=2>The History of Giles Gingerbread, 1d.
<!-- ... -->
<item =42>A Curious Collection of Travels,
selected from the Writers of all Nations,
10 Vol, Pr. bound 1l.
</list>
<div1 type=advert>
<head>By the KING's Royal Patent,</head>
<hgead>Are sold by J. NEWBERY, at the Bible and Sun in
St. Paul's Church-Yard.
<list>
<item n=1>Dr. James's Powders for Fevers, the
Small-Pox, Measles, Colds, &c. 2s. 6d
<item n=2>Dr. Hooper's Female Pills, 1s.
<!-- ... -->
</list></div1></back>
```

The content model of back matter is identical to that of front matter, reflecting the facts of cultural history.
Part: base tag set for common core features
Member of classes: declaring
DTD file: teiback2
Occurs within: text

May contain: argument byline div div1 docAuthor docDate epigraph epilogue head opener performance prologue salute set signed titlePage

Declaration:

```
<!ELEMENT back          - O  ( (%m.front)*, ( ( (%m.divtop),
                             (%m.divtop | titlePage)*) | (
                             (div), (div | (%m.front))*) | (
                             (div1), (div1 | (%m.front))*) )? )
                                                              >

<!ATTLIST back               %a.global
                             %a.declaring                     >
```

Discussed in: 7.6 ('Back Matter') on p. 205; 7 ('Default Text Structure') on p. 183

baseWsd

<baseWsd> (i.e. base writing system declaration) identifies a writing system declaration whose mappings among characters, forms, entity names, and bit patterns are to be incorporated (possibly with modifications) in this writing system declaration.

Attributes: [None: global and inherited attributes only.]

Reference in a WSD to a base WSD makes available the set of mappings among characters, forms, entity names, and bit patterns declared in the base WSD; the language and writing system (direction) information given in the base WSD is ignored.

Part: auxiliary tag set for writing system declarations

Member of classes: baseStandard

DTD file: teiwsd2

Data description: Empty.

Occurs within: characters

May contain: [none]

Declaration:

```
<!ELEMENT baseWsd       - O  EMPTY                            >

<!ATTLIST baseWsd            %a.global
                             %a.baseStandard                  >
```

Discussed in: 25.4.1 ('Base Components of the WSD') on p. 576

bibl

<bibl> (i.e. bibliographic citation) contains a loosely-structured bibliographic citation of which the sub-components may or may not be explicitly tagged.

Attributes: [None: global and inherited attributes only.]

Example:

```
<bibl>Blain, Clements and Grundy: Feminist Companion to Literature in
English (Yale, 1990)
</bibl>
```

Example:

```
<bibl><title.piece>The Interesting story of the Children in the
 Wood<title.piece>.
    In <author>Victor E Neuberg</author>,
    <title>The Penny Histories</title>.
    <publisher>OUP</publisher>
    <date>1968</date>.
</bibl>
```

Part: base tag set for common core features

Member of classes: bibl, declarable [and indirectly also:] common, inter

DTD file: teicore2

Data description: Contains either just prose, or any combination of elements from the *citation* class

Occurs within: add admin argument body broadcast camera caption case castList cell cit colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem listBibl

meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition scriptStmt seg set sic sound sourceDesc stage stress subc supplied syll tagUsage taxonomy tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref

May contain: #PCDATA abbr add address analytic anchor att author biblScope c caesura cl corr date dateRange dateStruct del distinct edition editor emph expan extent foreign formula gap gi gloss handShift hi idno imprint lang link m measure mentioned monogr name note num orig oRef oVar phr ptr publisher pubPlace pRef pVar ref reg respStmt rs s seg series sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT bibl          - O  (#PCDATA | %m.phrase; |
                              %m.biblPart;)*                  >

<!ATTLIST bibl                %a.global
                              %a.declarable                   >
```

Discussed in: 6.10.1 ('Elements of Bibliographic References') on p. 163; 5.2.7 ('The Source Description') on p. 90; 13.2 ('Tags for Terminological Data') on p. 312; 23.3.2 ('Declarable Elements') on p. 552

biblFull

<biblFull> contains a fully-structured bibliographic citation, in which all components of the TEI file description are present.

Attributes: [None: global and inherited attributes only.]

Example:

```
<biblFull>
        <titleStmt>
           <author>Blain, Virginia
           <author>Clements, Patricia
           <author>Grundy, Isobel
           <title>The Feminist Companion to Literature in English:
               women writers from the middle ages to the present
        </titleStmt>
        <editionStmt>
           <edition>UK edition</edition>
        <extent>1231 pp</extent>
        <publicationStmt>
          <creationDate>1990
          <publisher>Yale University Press
          <place>New Haven and London
          <date>1990
        </publicationStmt>
        <sourceDesc>No source: this is an original work</source.desc>
</biblFull>
```

Part: base tag set for common core features

Member of classes: bibl, declarable [and indirectly also:] common, inter

DTD file: teicore2

Data description:

Occurs within: add admin argument body broadcast camera caption case castList cell cit colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem listBibl meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition scriptStmt seg set sic sound sourceDesc stage stress subc supplied syll tagUsage taxonomy tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref

May contain: editionStmt extent notesStmt publicationStmt seriesStmt sourceDesc titleStmt

Declaration:

```
<!ELEMENT biblFull      - O  (titleStmt, editionStmt?, extent?,
                              publicationStmt, seriesStmt?,
                              notesStmt?, sourceDesc*)        >

<!ATTLIST biblFull            %a.global
                              %a.declarable                   >
```

Discussed in: 6.10.1 ('Elements of Bibliographic References') on p. 163; 5.2.7 ('The Source Description') on p. 90; 13.2 ('Tags for Terminological Data') on p. 312; 23.3.2 ('Declarable Elements') on p. 552

biblScope

<biblScope> (i.e. scope of citation) defines the scope of a bibliographic reference, for example as a list of pagenumbers, or a named subdivision of a larger work.

Attributes:

type identifies the type of information conveyed by the element, e.g. "pages", "volume".
Data type: CDATA
Sample values include:

volume the element contains a volume number.
issue the element contains an issue number, or volume and issue numbers.
pages the element contains a page number or page range.
chapter the element contains a chapter indication (number and/or title)
part the element identifies a part of a book or collection.

Default: #IMPLIED

Example:

```
<biblScope>pp 12-34
```

Example:

```
<biblScope>Volume 2 "My apprenticeship"
```

Part: base tag set for common core features
Member of classes: biblPart
DTD file: teicore2
Occurs within: bibl imprint monogr series
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT biblScope     - O  (%phrase.seq)                    >
<!ATTLIST biblScope          %a.global
          type               CDATA              #IMPLIED      >
```

Discussed in: 6.10.2.3 ('Imprint, Pagination, and Other Details') on p. 171

biblStruct

<biblStruct> (i.e. structured bibliographic citation) contains a structured bibliographic citation, in which only bibliographic subelements appear and in a specified order.

Attributes: [None: global and inherited attributes only.]
Example:

```
<biblStruct>
    <monogr>
         <author>Blain, Virginia
         <author>Clements, Patricia
         <author>Grundy, Isobel
         <title>The Feminist Companion to Literature in
                English: women writers from the middle ages
                to the present</>
         <imprint><publisher>Yale University Press</>
                  <city>New Haven and London</>
                  <date>1990</>
</monogr>
</biblStruct>
```

Part: base tag set for common core features
Member of classes: bibl, declarable [and indirectly also:] common, inter
DTD file: teicore2
Data description:

Occurs within: add admin argument body broadcast camera caption case castList cell cit colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem listBibl meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition scriptStmt seg set sic sound sourceDesc stage stress subc supplied syll tagUsage taxonomy tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref

May contain: analytic idno monogr note series

Declaration:

```
<!ELEMENT biblStruct    - O  (analytic?, (monogr, series*)+,
                              (note | idno)*)                    >

<!ATTLIST biblStruct         %a.global
                             %a.declarable                       >
```

Discussed in: 6.10.1 ('Elements of Bibliographic References') on p. 163; 5.2.7 ('The Source Description') on p. 90; 13.2 ('Tags for Terminological Data') on p. 312; 23.3.2 ('Declarable Elements') on p. 552

bicond

<bicond> (i.e. bi-conditional feature-structure constraint) defines a biconditional feature-structure constraint; both consequent and antecedent are specified as feature structures or groups of feature structures; the constraint is satisfied if both *subsume* a given feature structure, or if both do not.

Attributes: [None: global and inherited attributes only.]

Part: auxiliary tag set for feature system declarations

Member of classes:

DTD file: teifsd2

Data description: May contain two feature structures or feature-structure groups, separated by an empty <iff> element.

Occurs within: fsConstraints

May contain: f fs fAlt iff

Declaration:

```
<!ELEMENT bicond        - O  ((fs | f | fAlt), iff, (fs | f |
                              fAlt))                             >

<!ATTLIST bicond             %a.global                           >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

birth

<birth> (i.e. Birth details) contains information about a person's birth, such as its date and place.

Attributes:

date specifies the date of birth in an ISO standard form (yyyy-mm-dd).
Data type: %ISO-DATE
Value: a date in ISO standard form (yyyy-mm-dd).
Default: #IMPLIED

Example:

```
<birth>Before 1920, Midlands region.
```

Example:

```
<birth date=1960-12-10>In a small cottage near
<place>Aix-la-Chapelle</place>, early in the morning of <date>10 Dec 1960</date>
```

Dates and places, if included in the content of this element, should be tagged using the general <date> and <place> element.

Part: auxiliary tag set for corpora and collections

Member of classes: demographic

DTD file: teicorp2

Data description:

Occurs within: person personGrp

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT birth          - -  (%phrase.seq)                    >

<!ATTLIST birth               %a.global
          date                %ISO-date          #IMPLIED       >
```

Discussed in: 23.2.2 ('The Participants Description') on p. 545

bloc

<bloc> (i.e. bloc) a geo-political unit containing one or more nation states.

Attributes: [None: global and inherited attributes only.]

Example:

```
<bloc type='economic union'>the European Union</bloc>
<bloc type='continent'>Africa</bloc>
```

Part: additional tag set for

Member of classes: placePart [and indirectly also:] names

DTD file: teind2

Occurs within: placeName

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT bloc           - -  (%phrase.seq)                    >

<!ATTLIST bloc                %a.global
                              %a.placePart                     >
```

Discussed in: 20.2 ('Place Names') on p. 493

body

<body> (i.e. text body) contains the whole body of a single unitary text, excluding any front or back matter.

Attributes: [None: global and inherited attributes only.]

Part: base tag set for common core features

Member of classes: declaring

DTD file: teistr2

Occurs within: text

May contain: argument bibl biblFull biblStruct byline camera caption castList cit closer div div0 div1 docAuthor docDate entry entryFree epigraph event eTree graph head kinesic l label lg lg1 list listBibl move note opener p pause q quote salute shift signed sound sp stage superentry tech termEntry trailer tree u view vocal writing

Declaration:

```
<!ELEMENT body           - O  ((%m.divtop;)*, ( (div+ | div0+ |
                              div1+) | ( (%component)+, (div* |
                              div0* | div1*))), (%m.divbot;)*)  >

<!ATTLIST body                %a.global
                              %a.declaring                     >
```

Discussed in: 7 ('Default Text Structure') on p. 183; 13.4 ('Overall Structure of Terminological Documents') on p. 319

broadcast

<broadcast> (i.e. broadcast) describes a broadcast used as the source of a spoken text.

Attributes: [None: global and inherited attributes only.]

Example:

```
<broadcast>
     <bibl>
     <author>Radio Trent
     <title>Gone Tomorrow
     <resp><role>Presenter<name>Tim Maby
```

```
        <resp><role>Producer<name>Mary Kerr
        <date>12 June 89, 1230 pm
        </bibl>
    </broadcast>
```

Part: auxiliary tag set for TEI headers
Member of classes: declarable
DTD file: teihdr2
Data description:
Occurs within: recording
May contain: bibl biblFull biblStruct p recording
Declaration:

```
<!ELEMENT broadcast     - -  (p+ | bibl | biblStruct | biblFull
                              | recording)                      >

<!ATTLIST broadcast           %a.global
                              %a.declarable                     >
```

Discussed in: 5.2.9 ('Computer Files Composed of Transcribed Speech ') on p. 91; 23.3.2 ('Declarable Elements') on p. 552

<byline> contains the primary statement of responsibility given for a work on its title page or at the head or end of the work. byline

Attributes: [None: global and inherited attributes only.]
Example:

```
<byline>Written by a CITIZEN who continued all the
while in <place>London</place>. Never made publick before.</byline>
```

Example:

```
<byline>Written from her own MEMORANDUMS</byline>
```

Example:

```
<byline>By George Jones, Political Editor, in Washington
</byline>
```

Example:

```
<dateline>Zagreb:</dateline>
<byline>de notre envoy&eacute; sp&eacute;cial.</byline>
```

Example:

```
<byline>BY
<docAuth>THOMAS PHILIPOTT,</docAuth>
Master of Arts,
(Somtimes)
Of Clare-Hall in Cambridge.
</byline>
```

The byline on a title page may include either the name or a description for the document's author. Where the name is included, it may optionally be tagged using the <docAuth> element.
Part: base tag set for common core features
Member of classes: divbot, divtop, tpParts
DTD file: teistr2
Data description:
Occurs within: back body castList div div0 div1 div2 div3 div4 div5 div6 div7 epilogue front group lg performance prologue titlePage
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct docAuthor emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT byline        - O  (%phrase.seq; | docAuthor)*        >

<!ATTLIST byline              %a.global                         >
```

Discussed in: 7.2.2 ('Openers and Closers') on p. 192; 7.4 ('Front Matter') on p. 201

c

`<c>` (i.e. character) represents a character.

Attributes: [None: global and inherited attributes only.]

Example:

```
<c type=punctuation>?</c>
```

Part: additional tag set for

Member of classes: seg [and indirectly also:] phrase

DTD file: teiana2

Data description: Character data. Should only contain a single character or an entity that represents a single character.

Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale m measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view w wit witness witDetail writing xr xref

May contain: #PCDATA

Declaration:

```
<!ELEMENT c              - -  (#PCDATA)                    >
<!ATTLIST c                   %a.global
                              %a.seg                       >
```

Discussed in: 15.1 ('Linguistic Segment Categories') on p. 382

caesura

`<caesura>` marks the point at which a metrical line may be divided.

Attributes: [None: global and inherited attributes only.]

Example:

```
<l>Hw&ae;t we Gar-Dena <caesura> in gear-dagum
<l>&t;eod-cyninga <caesura> &t;rym gefrunon,
<l>hy &d;a &ae;&t;elingas <caesura> ellen fremedon.
.
```

Part: base tag set for verse texts

Member of classes: phrase.verse [and indirectly also:] phrase

DTD file: teivers2.dtd

Data description: Empty element

Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: [none]

Declaration:

```
<!ELEMENT caesura       - O  EMPTY                          >

<!ATTLIST caesura            %a.global                      >
```

Discussed in: 9.3 ('Components of the Verse Line') on p. 218

<camera> describes a particular camera angle or viewpoint in a screen play. camera

Attributes:

type characterizes the camera angle in some respect, e.g. as a close-up, medium shot, etc.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED

Example:

```
<view>George glances at the window--and freezes.
<camera>New angle--shock cut</camera>
Out the window the body of a dead man suddenly slams into
<hi>frame</hi>
<!-- ... -->
</view>
```

Part: base tag set for performance texts
Member of classes: stageDirection [and indirectly also:] comp.drama, inter
DTD file: teidram2
Data description: Contains character data and phrase level elements.
Occurs within: add admin argument body camera caption case castList cell colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT camera        - -  (%paraContent)                 >

<!ATTLIST camera             %a.global
          type               CDATA              #IMPLIED    >
```

Discussed in: 10.3.1 ('Technical Information') on p. 248; 10.3 ('Other Types of Performance Text') on p. 246

<caption> contains the text of a caption or other text displayed as part of a film script or screenplay. caption

Attributes: [None: global and inherited attributes only.]
Example:

```
<camera>Zoom in to overlay showing some stock film of hansom cabs
galloping past</camera>
<caption>London, 1895.</caption>
<caption>The residence of Mr Oscar Wilde.</caption>
<sound>Suitably classy music starts.</sound>
<view>Mix through to Wilde's drawing room. A crowd of suitably
dressed folk are engaged in typically brilliant conversation,
laughing affectedly and drinking champagne.</who>
<sp who=TJ><speaker>Prince of Wales</speaker>
<p>My congratulations, Wilde. Your latest play is a great success.
```

A specialized form of stage direction.
Part: base tag set for performance texts

Member of classes: stageDirection [and indirectly also:] comp.drama, inter
DTD file: teidram2
Data description: Contains character data and phrase level elements.
Occurs within: add admin argument body camera caption case castList cell colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT caption       - O  (%paraContent)                >
<!ATTLIST caption            %a.global                     >
```

Discussed in: 10.3.1 ('Technical Information') on p. 248; 10.3 ('Other Types of Performance Text') on p. 246

case

<case> contains grammatical case information given by a dictionary for a given form.

Attributes: [None: global and inherited attributes only.]
This element is synonymous with <gram type=case>.
Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, morphInfo
DTD file: teidict2
Data description: May contain character data and phrase-level elements. Typical values will be of the form "nom", "acc", "dat", "gen", etc.
Occurs within: etym trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT case          - -  (%paraContent;)               >
<!ATTLIST case               %a.global
                             %a.dictionaries               >
```

Discussed in: 12.3.1 ('Information on Written and Spoken Forms') on p. 279

castGroup

<castGroup> (i.e. Cast list grouping) groups one or more individual <castItem> elements within a cast list.

Attributes: [None: global and inherited attributes only.]
Example:

```
<castGroup rend=braced><head>friends of Mathias
   <castItem><role>Walter<actor>Mr Frank Hall
   <castItem><role>Hans<actor>Mr F.W. Irish
</castGroup>
```

Use the rend attribute to indicate whether the grouping is indicated by a brace, white space, font change, etc.
Part: base tag set for performance texts
Member of classes:
DTD file: teidram2
Data description:
Occurs within: castGroup castList
May contain: castGroup castItem head trailer
Declaration:

```
<!ELEMENT castGroup     - -  (head?, (castItem | castGroup)+,
                              trailer?)                        >

<!ATTLIST castGroup           %a.global                        >
```

Discussed in: 10.1.4 ('Cast Lists') on p. 233

<castItem> (i.e. Cast list item) contains a single entry within a cast list, describing either a single role or a list of non-speaking roles. castItem

Attributes:

type characterizes the cast item.
Data type: (ROLE | LIST)
Sample values include:
role the item describes a single role.
list the item describes a list of non-speaking roles.
Default: ROLE

Example:

```
<castItem><role>Player</><actor>Mr Milward</></castItem>
```

Example:

```
<castItem type=list>Constables, Drawer, Turnkey, etc.</castItem>
```

Part: base tag set for performance texts
Member of classes:
DTD file: teidram2
Data description: Contains character data with phrase-level elements, <role>, <roleDesc>, and <actor>.
Occurs within: castGroup castList
May contain: #PCDATA abbr actor add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg role roleDesc rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT castItem      - O  (role | roleDesc | actor |
                              (%phrase.seq))*                  >

<!ATTLIST castItem            %a.global
          type                (role | list)       role       >
```

Discussed in: 10.1.4 ('Cast Lists') on p. 233

<castList> (i.e. cast list) contains a single cast list or dramatis personae. castList

Attributes: [None: global and inherited attributes only.]
Example:

```
<castList>
<castGroup><head rend=braced>Mendicants</head>
  <castItem><role>Aafaa</><actor>Femi Johnson</></>
  <castItem><role>Blindman</><actor>Femi Osofisan</></>
  <castItem><role>Goyi</><actor>Wale Ogunyemi</></>
  <castItem><role>Cripple</><actor>Tunji Oyelana</></>
</castGroup>
<castItem>
  <role>Si Bero</><roleDesc>Sister to Dr Bero</>
  <actor>Deolo Adedoyin</>
</castItem>
<castGroup><head rend=braced>Two old women</head>
  <castItem><role>Iya Agba</><actor>Nguba Agolia</></>
  <castItem><role>Iya Mate</><actor>Bopo George</></>
</castGroup>
<castItem>
```

```
  <role>Dr Bero</><roleDesc>Specialist</>
  <actor>Nat Okoro</>
</castItem>
<castItem><role>Priest</><actor>Gbenga Sonuga</></>
<castItem><role>The old man</><roleDesc>Bero's father</>
   <actor>Dapo Adelugba</>
</castItem>
</castlist>
<stage>The action takes place in and around the home surgery of
Dr Bero, lately returned from the wars.</stage>
```

Part: base tag set for performance texts
Member of classes: comp.drama, inter
DTD file: teidram2
Data description:
Occurs within: add admin argument body camera caption case castList cell colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref
May contain: argument bibl biblFull biblStruct byline camera caption castGroup castItem castList cit docAuthor docDate entry entryFree epigraph event eTree graph head kinesic l label lg lg1 list listBibl move note opener p pause q quote salute shift signed sound sp stage superentry tech termEntry tree u view vocal writing
Declaration:

```
<!ELEMENT castList      - -  ((%m.divtop)*, (%component)*,
                              (castItem | castGroup)+,
                              (%component)*)                    >

<!ATTLIST castList           %a.global                          >
```

Discussed in: 10.1.4 ('Cast Lists') on p. 233; 10.1 ('Front and Back Matter ') on p. 228

catDesc

<catDesc> (i.e. category description) describes some category within a taxonomy or text typology, either in the form of a brief prose description or in terms of the situational parameters used by the TEI formal <textDesc>.

Attributes: [None: global and inherited attributes only.]
Example:

```
      <catDesc>Prose reportage
```

Example:

```
<catDesc>
    <textDesc n=novel>
     <channel mode=w>print; part issues
     <constitution type=single>
     <derivation type=original>
     <domain type=art><factuality type=fiction>
     <interaction type=none>
     <preparedness type=prepared>
     <purposes><purpose type=entertain degree=high>
              <purpose type=inform degree=medium>
     </purposes>
    </textDesc>
</catedesc>
```

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Occurs within: category
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr

ptr pRef pVar ref reg rs s seg sic soCalled tag term textDesc time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT catDesc       - O  (%phrase.seq; | textDesc)          >
<!ATTLIST catDesc            %a.global                          >
```

Discussed in: 5.3.6 ('The Classification Declaration') on p. 104

<category> (i.e. category) contains an individual descriptive category, possibly nested within a superordinate category, within a user-defined taxonomy. category

Attributes: [None: global and inherited attributes only.]

Example:

```
<category id=B1><catDesc>Prose reportage</category>
```

Example:

```
<category id=B1><catDesc>Prose
<category id=B11><catDesc>reportage</category>
<category id=B12><catDesc>fiction</category>
</category>
```

Part: auxiliary tag set for TEI headers

Member of classes:

DTD file: teihdr2

Occurs within: category taxonomy

May contain: category catDesc

Declaration:

```
<!ELEMENT category      - -  (catDesc, category*)               >
<!ATTLIST category           %a.global                          >
```

Discussed in: 5.3.6 ('The Classification Declaration') on p. 104

<catRef> (i.e. category reference) specifies one or more defined categories within some taxonomy or text typology. catRef

Attributes:

target identifies the categories concerned
Data type: IDREFS
Value: One or more identifiers for <category> elements defined in the current document.
Default: #REQUIRED

scheme identifies the classification scheme within which the set of categories concerned is defined
Data type: IDREF
Value: identifier of the associated <taxonomy> element.
Default: #IMPLIED

Example:

```
<catRef scheme= S1 target="BA C1">
```

The scheme attribute need be supplied only if more than one taxonomy has been declared

Part: auxiliary tag set for TEI headers

Member of classes:

DTD file: teihdr2

Data description:

Occurs within: textClass

May contain: [none]

Declaration:

```
<!ELEMENT catRef        - O  EMPTY                              >
<!ATTLIST catRef             %a.global
          target             IDREFS              #REQUIRED
          scheme             IDREF               #IMPLIED      >
```

Discussed in: 5.4.3 ('The Text Classification') on p. 111

cb

<cb> (i.e. column break) marks the boundary between one column of a text and the next in a standard reference system.

Attributes:

ed (i.e. edition) indicates the edition or version in which the column break is located at this point
Data type: CDATA
Value: Any string of characters; usually a siglum conventionally used for the edition.
Default: #IMPLIED
Example:

```
<cb ed=Riverside n=123>
```

n (i.e. number or name) indicates the number or other value associated with the column which follows the point of insertion of this <cb> element.
Data type: CDATA
Value: Any string of characters.
Default: #IMPLIED
Encoders should adopt a clear and consistent policy as to whether the numbers associated with column breaks relate to the physical sequence number of the column in the whole text, or whether columns are numbered within the page. By convention, the <cb> element is placed at the head of the column to which it refers.

Like other forms of milestone tag, <cb> tags cannot be automatically verified by SGML; for better validation, a concurrent markup stream should be used.

Part: additional tag set for common core features
Member of classes: refsys
DTD file: teicore2
Occurs within: [none]
May contain: [none]
Declaration:

```
<!ELEMENT cb            - O  EMPTY                          >

<!ATTLIST cb                 %a.analysis
                             %a.linking
                             %a.terminology
          id                 ID                 #IMPLIED
          lang               IDREF              %INHERITED

          rend               CDATA              #IMPLIED
          ed                 CDATA              #IMPLIED
          n                  CDATA              #IMPLIED    >
```

Discussed in: 6.9.3 ('Milestone Tags') on p. 158

cell

<cell> contains one cell of a table.

Attributes:

role indicates the kind of information held in the cell.
Data type: CDATA
Sample values include:

label labelling or descriptive information only.

data data values.

Default: DATA
The value specified overrides any default specified by the role attribute of the parent <row> element, for this cell only.

rows indicates the number of rows occupied by this cell.
Data type: NUMBER
Value: A number; a value greater than one indicates that this cell spans several rows.

Default: 1

Where several cells span several rows, it may be more convenient to use nested tables.

cols indicates the number of columns occupied by this cell.

Data type: NUMBER

Value: A number; a value greater than one indicates that this cell spans several columns.

Default: 1

Where an initial cell spans an entire row, it should be treated as a heading.

Example:

```
<row>
  <cell role=label>General conduct</cell>
  <cell role=data>Not satisfactory, on account of his great
       unpunctuality and inattention to duties</cell>
</row>
```

Part: additional tag set for formulae

Member of classes:

DTD file: teifig2

Data description:

Occurs within: row

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref

Declaration:

```
<!ELEMENT cell          - O  (%paraContent)                  >

<!ATTLIST cell               %a.global
          role               CDATA              data
          rows               NUMBER             1
          cols               NUMBER             1            >
```

Discussed in: 22.1.1 ('The TEI Table DTD') on p. 524

<certainty> indicates the degree of certainty or uncertainty associated with some aspect of the text markup. certainty

Attributes:

target points at the elements whose markup is uncertain.

Data type: IDREFS

Value: one or more SGML ID values.

Default: #REQUIRED

Example:

```
Elizabeth went to <persName id=p1>Essex</persName>
<certainty target=p1 locus=gi degree='0.6'>
```

If more than one IDREF is given, the <certainty> element is interpreted as applying to all. If no ID is present on the element being annotated, the attribute should give the ID of a <ptr> element which points at the element being annotated; for further discussion of this indirect pointing mechanism, see chapter 14 ('Linking, Segmentation, and Alignment') on p. 331.

locus indicates the precise location of the uncertainty in the markup: applicability of the element, precise position of the start- or end-tag, value of a specific attribute, etc.

Data type: CDATA

Sample values include:

#gi uncertain whether the element used actually applies to the passage.

#startloc start-tag may not be correctly located.

#endloc end-tag may not be correctly located.

#location both the start-tag and the end-tag may not be correctly located.

name the value given for the attribute name is uncertain.

#transcribedContent the content of the element may not be a correct transcription of the source text.

#suppliedContent the content of the element may not have been correctly supplied by the reader, e.g. as in the cases of <corr> and <abbrev> elements.

Default: #REQUIRED

The "#" distinguishes the terms of the controlled vocabulary from possible collisions with attribute names. Extensions to this vocabulary should also use this prefix.

assertedValue provides an alternative value for the aspect of the markup in question—an alternative generic identifier, transcription, or attribute value, or the ID of an <anchor> element (to indicate an alternative starting or ending location). If an assertedValue is given, the confidence level specified by degree applies to the alternative markup specified by assertedValue; if none is given, it applies to the markup in the text.

Data type: CDATA

Value: generic identifier, attribute value, location (e.g. indicated by an IDREF to an <anchor> element or to an <xptr> element), or other appropriate alternative value.

Default: #IMPLIED

Example:

```
Elizabeth went to <persName id=p1>Essex</persName>
<certainty target=p1 locus='#gi'
           assertedValue='place' degree='0.2'>
```

This attribute makes it possible to indicate the degree of confidence in a specific alternative to some aspect of the markup. In the example above the encoder is expressing the likelihood (.2) that the generic identifier should be <place> rather than <persName>, which is the coded element.

desc further describes the uncertainty in prose, perhaps indicating its nature, cause, or the justification for the degree of confidence asserted.

Data type: CDATA

Value: a prose description of how and why the markup is uncertain.

Default: #IMPLIED

Example:

```
Elizabeth went to <persName id=p1>Essex</persName>
<certainty target=p1 locus='#gi'  degree='0.2'
desc='As this was written in the Spring
it is probably the Earl, not the town'>
```

In a given project, it may be possible to enumerate a finite list of recognized types and causes of uncertainty; in such cases, it will be useful to control the vocabulary used in this attribute, to aid later mechanical manipulation. It is not possible to suggest such a controlled vocabulary for general use.

given indicates conditions assumed in the assignment of a degree of confidence.

Data type: CDATA

Value: a characterization of the conditions which are assumed in the assignment of a degree of confidence. This may be in prose.

Default: #IMPLIED

Example:

```
<!-- in the header, hand 'h1' is
    identified as that of 'msm' ... -->
<hand id=h1 scribe=msm>

 <!-- in the text, the scribe has corrected 'Wessex'
     into 'Essex' ... -->
Elizabeth went to <corr id=c1 sic='Wessex' resp=msm>Essex</corr>.

<!-- we are 60% certain that hand h1 is 'msm',
     and 90% certain that if h1 is 'msm', then
     it is 'msm' who corrected 'Wessex'
     into 'Essex'.  -->
```

```
<certainty id=h1 target=h1 locus=scribe degree='0.6'>
<certainty target=p1 locus=resp given=h1 degree='0.9'>
```

A project may wish to control the vocabulary used in this attribute.

The envisioned typical value of this attribute would be the identifier of another <certainty> element or a list of such identifiers. It may thus be possible to construct probability networks by chaining <certainty> elements together. Such networks would ultimately be grounded in unconditional <certainty> elements (with no value for given). The semantics of this chaining would be understood in this way: if a <certainty> element is specified, via an IDREF, as the assumption, then it is not the attribution of uncertainty that is the assumption, but rather the assertion itself. For instance, in the example above, the first <certainty> element indicates that the confidence in the identification of the new scribe as "msm". The second indicates the degree of confidence that "Essex" is a personal name, given that the new scribe is "msm". Note that the given in the second <certainty> element is not the assertion that the likelihood that msm is the new scribe is 0.6, but simply the assertion that msm is the new scribe; this is a recommended convention to facilitate building networks.

The ambitious encoder may wish to attempt complex networks or probability assertions, experimenting with references to other elements or prose assertions, and deploying feature structure connectives such as <alt>, <join>, and <not>. However, we do not believe that the <certainty> element gives, at this time, a comprehensive ambiguity-free system for indicating certainty.

degree indicates the degree of confidence assigned to the aspect of the markup named by the locus attribute.

Data type: CDATA

Value: Values of degree might be yes or no, the reals between 0 and 1, or traditional characterizations such as "doubtful", "circa", etc. Generally we recommend decimal numbers between 0 and 1, where larger numbers denote a greater degree of confidence in the assertions; 0 representing "certainly false" and 1 representing "certainly true".

Default: #IMPLIED

For discussion of this example, see section 17.1.2 ('Structured Indications of Uncertainty') on p. 436.

Example:

```
Earnest went to <anchor id=a1> old
<persName id=p1>Saybrook</persName>.

<certainty id=c1 target=p1 locus='#gi' degree='0.6'>
<certainty given=c1 target=p1 locus=startloc degree='0.9'>

<certainty id=c2 target=p1 locus='#gi'
           assertedValue='persName' degree='0.4'>
<certainty given=c2 target=p1 locus=startloc degree='0.5'>
<certainty id=c3
           given=c1 target=p1 locus=startloc
           assertedValue='a1' degree='0.5' >
```

Part: additional tag set for certainty
Member of classes: metadata [and indirectly also:] globincl
DTD file: teicert2
Data description: Empty.
Occurs within: [none]
May contain: [none]
Declaration:

```
<!ELEMENT certainty     - O  EMPTY                              >

<!ATTLIST certainty          %a.global
          target             IDREFS              #REQUIRED
          locus              CDATA               #REQUIRED
```

```
assertedValue      CDATA              #IMPLIED
desc               CDATA              #IMPLIED
given              CDATA              #IMPLIED
degree             CDATA              #IMPLIED      >
```

Discussed in: 17.1.2 ('Structured Indications of Uncertainty') on p. 436

change

`<change>` summarizes a particular change or correction made to a particular version of an electronic text which is shared between several researchers.

Attributes: [None: global and inherited attributes only.]

Example:

```
<change n="P2.2">
   <date>21 Dec 91 <respStmt><name>LB</><resp>ed.</></respStmt>
   <item>Added examples to section 3</>
</change>
<change>
   <date>11 Nov 91 <respStmt><name>LB</><resp>ed.</></respStmt>
   <item>Deleted chapter 10</>
</change>
```

Changes should be recorded in a consistent order, for example with the most recent first.

Part: auxiliary tag set for TEI headers

Member of classes:

DTD file: `teihdr2`

Data description:

Occurs within: revisionDesc

May contain: date item respStmt

Declaration:

```
<!ELEMENT change        - O  (date, respStmt+, item)          >
<!ATTLIST change             %a.global                        >
```

Discussed in: 5.5 ('The Revision Description') on p. 112

channel

`<channel>` (i.e. primary channel) describes the medium or channel by which a text is delivered or experienced. For a written text, this might be print, manuscript, e-mail, etc.; for a spoken one, radio, telephone, face-to-face, etc.

Attributes:

mode specifies the mode of this channel with respect to speech and writing.

Data type: (S | W | WS | SW | M | X)

Sample values include:

s spoken

w written

sw spoken to be written (e.g. dictation)

ws written to be spoken (e.g. a script)

m mixed modes

x unknown or inapplicable

Default: X

Example:

```
<channel mode=s>face-to-face conversation
```

Part: auxiliary tag set for corpora and collections

Member of classes:

DTD file: `teicorp2`

Data description:

Occurs within: textDesc

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT channel       - O  (%phrase.seq)                    >

<!ATTLIST channel            %a.global
          mode               (s | w | ws | sw | m | x)
                                                   x         >
```

Discussed in: 23.2.1 ('The Text Description') on p. 541

character

<character> defines one unit in a writing system, supplementing or overriding information provided in the base coded character sets, writing system declarations, and entity sets.

Attributes:

class describes the function of the character using a prescribed classification.

Data type: (LEXICAL | PUNC | LEXPUNC | DIGIT | SPACE | DL | LD | DIA | JOINER | OTHER)

Sample values include:

lexical character is used in writing words (lexical items) of the language (includes members of syllabaries and ideographic systems, as well as composite letter-plus-diacritic combinations)

punc character is a punctuation mark which does not appear within lexical items

lexpunc character can appear as a normal punctuation mark, but can also appear within a lexical item (and should usually, when occurring between two lexical characters, be treated as lexical—in English, hyphen and apostrophe are typically treated as members of this class)

digit character is an Arabic decimal numeral (0, 1, ... 9) (does not include superscript numbers, circled numbers, numeric dingbats, etc.)

space character represents some form of white space (space character, horizontal or vertical tab, newline, etc.)

dl character is a diacritic applying to the following lexical character

ld character is a diacritic applying to the preceding lexical character

dia character is a diacritic which is explicitly joined to a lexical character by a joiner character

joiner character is used to join a diacritic to the lexical character to which it applies (in some encoding schemes, the backspace control character may be used as a joiner; in others, a graphic character is used for the same function)

other character does not fall into any of the other classes (dingbats and other unusual characters fall here)

Default: LEXICAL

The classification of characters provided by this attribute serves both informative and normative purposes: it helps identify the character being described, and the classification is used to define the meaning of the special character-class codes in the TEI extended pointer syntax described in chapter 14 ('Linking, Segmentation, and Alignment') on p. 331.

The notion of "characters" as units in a writing system is widely spread, but not consistently defined; the <character> element should be used to identify whatever units the encoder wishes to distinguish as the meaningfully distinct graphic units of the writing system. In most cases, these will correspond to the units of coded character sets, but that this is not a requirement: a-umlaut, for example, may be treated as one character or two, depending on the user's preference, regardless of how the coded character set in use treats it. In most cases, also, the units distinguished by the <character> element will be the "graphemic" units of the writing system in question; however, since experts disagree on whether items like umlaut (let alone a given set of Chinese characters with regional variations in China, Korea, and Japan) are best treated as distinct graphemes or not, the association of <character> elements with the graphemes of a writing system provides at most a heuristic device for making reasonable decisions, rather than a definitive unambiguous test.

Different forms of the same "character" may be distinguished for whatever reason, as in the three-R example of chapter 4 ('Characters and Character Sets') on p. 71. In this case the

different letter forms are distinguished by documenting them in different <form> elements; the fact that the different letter shapes do not make a lexical difference in the text may be expressed by grouping all three letter forms under the same <character> element. (Alternatively, the three forms may be treated as three distinct characters, for convenience or for whatever reason, by defining a distinct <character> element for each.)
Part: auxiliary tag set for writing system declarations
Member of classes:
DTD file: teiwsd2
Data description: May contain one or more description elements (optional), a series of one or more <form> elements identifying different forms of the character, and an optional series of notes.
Occurs within: exceptions
May contain: desc form note
Declaration:

```
<!ELEMENT character     - O  (desc*, form+, note*)              >
<!ATTLIST character          %a.global
          class              (lexical | punc | lexpunc | digit
                             | space | DL | LD | dia | joiner |
                             other)              lexical       >
```

Discussed in: 25.4.2 ('Exceptions in the WSD') on p. 577

characters

<characters> contains a specification of the characters used in a particular writing system to write a particular language, and of how those characters are represented in electronic form.
Attributes: [None: global and inherited attributes only.]
Example:

```
<writingSystemDeclaration
    name='-//TEI P2: 1993//WSD ISO 8859-1//en'
    date='1993-06-01'>
    <language iso639=''>various</>
    <direction chars=LR lines=TB>
    <characters>
         <codedCharSet name='ISO 8859-1: 1992' authority='ISO'>
         <exceptions>
              <!-- ... -->
         </exceptions>
    </characters>
</writingSystemDeclaration>
```

Part: auxiliary tag set for writing system declarations
Member of classes:
DTD file: teiwsd2
Data description: May contain a series of base components (coded character sets, writing system declarations, and entity sets) followed by a set of exceptions to define differences from the base components.
Occurs within: writingSystemDeclaration
May contain: baseWsd codedCharSet entitySet exceptions
Declaration:

```
<!ELEMENT characters    - O  (codedCharSet*, baseWsd*,
                              entitySet*, exceptions?)          >
<!ATTLIST characters         %a.global                          >
```

Discussed in: 25.4.1 ('Base Components of the WSD') on p. 576; 25.1 ('Overall Structure of Writing System Declaration') on p. 571

children

<children> lists the elements which this element may directly contain.
Attributes: [None: global and inherited attributes only.]
Example:

```
<children>emph, hi, name, address, #PCDATA
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: the list may contain a description or a list of names separated by commas or spaces.
Occurs within: tagDoc
May contain: #PCDATA
Declaration:

```
<!ELEMENT children      - O  (#PCDATA)                          >

<!ATTLIST children           %a.global                          >
```

Discussed in: 27.1 ('The TagDoc Documentation Element') on p. 603

<cit> A quotation from some other document, together with a bibliographic reference to its source. cit

Attributes: [None: global and inherited attributes only.]
Example:

```
<cit>
<quote>and the breath of the whale is frequently
attended with such an insupportable smell, as to
bring on disorder of the brain.
</quote>
<bibl>Ulloa's South America</bibl>
</cit>
```

Part: base tag set for common core features
Member of classes: hqinter [and indirectly also:] common, inter
DTD file: teicore2
Data description: Must contain a single quote and a single bibliographic reference in either order
Occurs within: add admin argument body camera caption case castList cell colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition eg emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref
May contain: bibl biblFull biblStruct link ptr q quote ref xptr xref
Declaration:

```
<!ELEMENT cit           - -  ((q | quote) & (%m.bibl; |
                             %m.loc;))                       >

<!ATTLIST cit                %a.global                       >
```

Discussed in: 6.3.3 ('Quotation') on p. 127; 7.3 ('Groups of Texts') on p. 195; 12.3.5.1 ('Examples') on p. 290

<cl> (i.e. clause) represents a grammatical clause. cl

Attributes: [None: global and inherited attributes only.]
Example:

```
<cl type=relative function='clause modifier'>
Which frightened both the heroes so,
<lb><cl>They quite forgot their quarrel.</cl></cl>
```

Part: additional tag set for
Member of classes: seg [and indirectly also:] phrase
DTD file: teiana2
Data description: May contain character data and phrase-level elements.

Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT cl            - -  (%phrase.seq)                    >

<!ATTLIST cl                 %a.global
                             %a.seg                           >
```

Discussed in: 15.1 ('Linguistic Segment Categories') on p. 382

class

<class> specifies the name of an element class.

Attributes: [None: global and inherited attributes only.]

Example:

```
<classDoc>
 <class>phrase</class>
 <!-- ... -->
</classDoc>
```

Part: auxiliary tag set for tag set declarations

Member of classes:

DTD file: teitsd2

Data description: In TEI compatible encoding systems, the class name should be chosen so that a valid SGML name may be made from it. For attribute classes, this name is made by prefixing the class name with the string "a."; for model classes, the string "m." is used.

Occurs within: classDoc

May contain: #PCDATA

Declaration:

```
<!ELEMENT class         - O  (#PCDATA)                        >

<!ATTLIST class              %a.global                        >
```

Discussed in: 27.2 ('Element Classes') on p. 606

classCode

<classCode> (i.e. classCode) contains the classification code used for this text in some standard classification system.

Attributes:

scheme identifies the classification system or taxonomy in use.
Data type: IDREF
Value: must identify a <taxonomy> element.
Default: #IMPLIED

Example:

```
<classCode scheme=DDC12>410
```

Part: auxiliary tag set for TEI headers

Member of classes:

DTD file: teihdr2

Occurs within: textClass
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT classCode      - -  (%phrase.seq)                    >

<!ATTLIST classCode           %a.global
          scheme              IDREF              #IMPLIED      >
```

Discussed in: 5.4.3 ('The Text Classification') on p. 111

<classDecl> (i.e. classification declarations) contains one or more taxonomies defining any classificatory codes used elsewhere in the text. classDecl

Attributes: [None: global and inherited attributes only.]
Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Occurs within: encodingDesc
May contain: taxonomy
Declaration:

```
<!ELEMENT classDecl      - -  (taxonomy+)                      >

<!ATTLIST classDecl           %a.global                        >
```

Discussed in: 5.3.6 ('The Classification Declaration') on p. 104; 5.3 ('The Encoding Description') on p. 93

<classDoc> contains reference information for one element class; either elements which appear together in SGML content models, or elements which share some common attribute. classDoc

Attributes:

type indicates whether this is an model class, an attribute class, or both.
Data type: (MODEL | ATTS | BOTH)
Sample values include:

model members of this class appear in the same SGML content models

atts members of this class share common attributes

both members of this class share attributes and appear in the same SGML content models

Default: #IMPLIED

Example:

```
<classDoc type=model ID=seg>
<class>seg</class>
<desc>elements for arbitrary segmentation.</desc>
<ptr target=COse>
</classDoc>
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description:
Occurs within: tsd
May contain: attList class classes desc equiv files part ptr remarks rs
Declaration:

```
<!ELEMENT classDoc       - O  (class, rs?, desc, attList?,
                               remarks?, part?, classes?, files?,
                               ptr*, equiv*)                 >
```

```
<!ATTLIST classDoc          %a.global
          type              (model | atts | both)
                                                  #IMPLIED      >
```

Discussed in: 27.2 ('Element Classes') on p. 606; 27 ('Tag Set Documentation') on p. 601

classes

<classes> specifies all the classes of which the documented element or class is a member or subclass.

Attributes:

names lists the identifiers of all classes of which the documented element or class is a member or subclass, possibly using parentheses to indicate inheritance.
Data type: CDATA
Value: a list of class names separated by spaces or commas, and optionally enclosed by parentheses; each name should be the class name specified for some element class in the scheme being documented or modified.
Default: #REQUIRED

Example:

```
<classes names='hqinter (inter (common))'>hqinter
[and indirectly also:] common, inter</classes>
```

This <classes> element indicates that the element documented is a member of the class *hqinter*, which is itself a subclass of the classes *inter* and thus also of *common*. The value of the names attribute and the content are synonymous, two ways of representing the same information.

Example:

```
<classes names='hqinter'>hqinter</classes>
```

This <classes> element indicates that the element documented is a member of the class *hqinter*, but gives no indication that *hqinter* is itself a subclass of *phrase*.

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: Empty
Occurs within: classDoc tagDoc
May contain: #PCDATA
Declaration:

```
<!ELEMENT classes       - O  (#PCDATA)                        >
<!ATTLIST classes           %a.global
          names             CDATA               #REQUIRED     >
```

Discussed in: 27.1 ('The TagDoc Documentation Element') on p. 603; 27.2 ('Element Classes') on p. 606

closer

<closer> groups together dateline, byline, salutation, and similar phrases appearing as a final group at the end of a division, especially of a letter.

Attributes: [None: global and inherited attributes only.]

Example:

```
...perhaps you will favour me with a sight
of it when convenient.</p>
<closer><salute>I remain, &amp;c. &amp;c.</salute>
 <signed>H. Colburn</signed>
</closer>
</div>
```

Example:

```
...and his heart was going like mad and yes
I said yes I will Yes.</p>
<closer>
 <dateline>
   <rs>Trieste-Z&uuml;rich-Paris,</rs>
```

```
    <date>1914-1921</date>
   </dateline>
  </closer>
  </div>
  </text>
```

Part: base tag set for common core features
Member of classes: divbot
DTD file: teistr2
Occurs within: body div div0 div1 div2 div3 div4 div5 div6 div7 epilogue group lg performance prologue
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateline dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s salute seg sic signed soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT closer        - O  (signed | dateline | salute |
                              %phrase.seq;)*                  >

<!ATTLIST closer             %a.global                        >
```

Discussed in: 7.2.4 ('Content of Textual Divisions') on p. 194; 7.2 ('Elements Common to All Divisions') on p. 190

<codedCharSet> (i.e. base coded character set) identifies a public or private coded character set which is used as a basic component of a writing system declaration. codedCharSet

Attributes: [None: global and inherited attributes only.]

Reference to a coded character set makes the set of bit-pattern-to-character mappings defined in the coded character set available for use in text encoded with the writing system declaration. Unless the character inventory and mappings are modified by the <exceptions> element, any character in any base coded character set may be used with its standard meaning within text to which the writing system declaration applies.

In order to make the mappings of bit patterns to characters more explicit than they sometimes are in character-set standards, the TEI will provide standard writing system declarations which document the bit-pattern-to-character mappings defined by various commonly used coded character sets.

Part: auxiliary tag set for writing system declarations
Member of classes: baseStandard
DTD file: teiwsd2
Data description: An empty element.
Occurs within: characters
May contain: [none]
Declaration:

```
<!ELEMENT codedCharSet  - O  EMPTY                            >

<!ATTLIST codedCharSet       %a.global
                             %a.baseStandard                  >
```

Discussed in: 25.4.1 ('Base Components of the WSD') on p. 576

<col> (i.e. column) contains one column of a multi-column reference edition. col

Attributes: [None: global and inherited attributes only.]
Example:

```
<(La)page n=32>
<(La)col n=a> ...
<(La)col n=b> ...
<(La)page n=33> ...
```

The <col> tag should be used wherever a standard reference scheme uses references to individual columns of a reference edition; otherwise it need not be used.
Part: base tag set for

Member of classes:
DTD file: teipl2
Data description: May contain <line> elements, or character data.
Occurs within: page
May contain: #PCDATA line
Declaration:

```
<!ELEMENT col           - O  (#PCDATA | line)*                  >

<!ATTLIST col                %a.global                          >
```

Discussed in: 31.6 ('Concurrent Markup for Pages and Lines') on p. 637

colloc

<colloc> (i.e. collocate) contains a collocate of the headword.

Attributes:

type classifies the collocation, using any convenient typology.
Data type: CDATA
Value: any string of characters, e.g. "preposition".
Default: #IMPLIED

Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, gramInfo
DTD file: teidict2
Data description: May contain character data and phrase-level elements.
Occurs within: gramGrp trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT colloc        - O  (%paraContent;)                    >

<!ATTLIST colloc             %a.global
                             %a.dictionaries
              type           CDATA              #IMPLIED       >
```

Discussed in: 12.3.2 ('Grammatical Information') on p. 284

cond

<cond> (i.e. conditional feature-structure constraint) defines a conditional feature-structure constraint; the consequent and the antecedent are specified as feature structures or feature-structure groups; the constraint is satisfied if both the antecedent and the consequent *subsume* a given feature structure, or if the antecedent does not.

Attributes: [None: global and inherited attributes only.]
Part: auxiliary tag set for feature system declarations
Member of classes:
DTD file: teifsd2
Data description: May contain an antecedent feature structure, an empty <then> element, and a consequent feature structure.
Occurs within: fsConstraints
May contain: f fs fAlt then
Declaration:

```
<!ELEMENT cond          - O  ((fs | f | fAlt), then, (fs | f |
                             fAlt))                             >

<!ATTLIST cond               %a.global                          >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

constitution

<constitution> describes the internal composition of a text or text sample, for example as fragmentary, complete, etc.

Attributes:

type specifies how the text was constituted.
Data type: (SINGLE | COMPOSITE | FRAGS | UNKNOWN)
Sample values include:
single a single complete text
composite a text made by combining several smaller items, each individually complete
frags a text made by combining several smaller, not necessarily complete, items
unknown composition unknown or unspecified
Default: SINGLE

Example:

```
<constitution type=frags>Prologues only.
```

The function of this element seems to overlap with both the ORG attribute on DIVs and the samplingDecl in the encodingDesc.

Part: auxiliary tag set for corpora and collections
Member of classes:
DTD file: teicorp2
Data description: contains any sequence of phrase level data
Occurs within: textDesc
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT constitution  - O  (%phrase.seq)                    >
<!ATTLIST constitution       %a.global
          type               (single | composite | frags |
                             unknown)            single        >
```

Discussed in: 23.2.1 ('The Text Description') on p. 541

<corr> (i.e. correction) contains the correct form of a passage apparently erroneous in the copy text. corr

Attributes:

sic gives the original form of the apparent error in the copy text.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED
Example:

```
for his nose was as sharp as a pen, and
<corr sic="a Table"> a' babbled</corr>
of green fields.
```

resp (i.e. responsibility) signifies the editor or transcriber responsible for suggesting the correction held as the content of the <corr> element.
Data type: CDATA
Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing, or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).
Default: %INHERITED
If the correction was made in the source, this attribute should be used to identify the hand of the corrector.

cert (i.e. certainty) signifies the degree of certainty ascribed to the correction held as the content of the <corr> element.
Data type: CDATA
Default: #IMPLIED

If all that is desired is to call attention to the fact that the copy text has been corrected, no attributes are required:

Example:

```
I don't know, Juan.  It's so far in the past now &mdash;
how <corr>can we</corr> prove or disprove anyone's theories?
```

It is also possible to provide a correct reading and to identify the individual responsible for the correction:

Example:

```
I don't know, Juan.  It's so far in the past now &mdash;
how <corr sic='we can' resp='MSM'>can we</corr> prove or
disprove anyone's theories?
```

The <corr> tag is a mirror of <sic>: the latter leaves the original text untouched, giving the correction as an attribute value; the former substitutes the correction, leaving the original reading as an attribute value. The choice between them is up to the encoder.

Part: additional tag set for common core features

Member of classes: edit [and indirectly also:] phrase

DTD file: teicore2

Data description: May contain character data and phrase-level elements.

Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph entry entryFree event expan eTree figure foreign formula gap gi gloss graph handShift hi kinesic l label lang lg lg1 link list listBibl m measure mentioned move name note num orig oRef oVar p pause phr ptr pRef pVar q quote ref reg rs s seg shift sic sound soCalled sp stage superentry table tag tech term termEntry text time timeRange timeStruct title tree u val view vocal w writing xptr xref TEI...end

Declaration:

```
<!ELEMENT corr          - -  (%specialPara;)                    >

<!ATTLIST corr              %a.global
          sic               CDATA               #IMPLIED
          resp              CDATA               %INHERITED

          cert              CDATA               #IMPLIED      >
```

Discussed in: 6.5.1 ('Correction of Apparent Errors') on p. 141

correction

<correction> (i.e. correction principles) states how and under what circumstances corrections have been made in the text.

Attributes:

status indicates the degree of correction applied to the text.

Data type: (HIGH | MEDIUM | LOW | UNKNOWN)

Sample values include:

high the text has been thoroughly checked and proofread.

medium the text has been checked at least once.

low the text has not been checked.

unknown the correction status of the text is unknown.

Default: UNKNOWN

method indicates the method adopted to indicate corrections within the text.
Data type: (SILENT | TAGS)
Sample values include:
silent corrections have been made silently
tags corrections have been represented using editorial tags
Default: SILENT

Example:

```
<correction><p>Errors in transcription controlled by using the
WordPerfect spelling checker, with a user defined dictionary of 500
extra words taken from Chambers Twentieth Century Dictionary.
```

May be used to note the results of proof reading the text against its original, indicating (for example) whether discrepancies have been silently rectified, or recorded using the editorial tags described in section 6.5 ('Simple Editorial Changes') on p. 140.

Part: auxiliary tag set for TEI headers
Member of classes: declarable
DTD file: teihdr2
Data description:
Occurs within: editorialDecl
May contain: p
Declaration:

```
<!ELEMENT correction    - O  (p+)                                  >
<!ATTLIST correction         %a.global
                             %a.declarable
          status             (high | medium | low | unknown)
                                                    unknown
          method             (silent | tags)      silent        >
```

Discussed in: 5.3.3 ('The Editorial Practices Declaration') on p. 96; 23.3.2 ('Declarable Elements') on p. 552

country

<country> (i.e. country) in an address, gives the name of the nation, country, colony, or commonwealth; in a place name given as a hierarchy of geo-political units, the <country> is larger or administratively superior to the <region> and smaller than the <bloc>.

Attributes: [None: global and inherited attributes only.]
Example:

```
<country n=DK>Denmark</>
```

If attributes are used to identify the country referred to (as the n is used above), the values should be taken from recognized lists of abbreviations, e.g. ISO 3166.

Part: base tag set for common core features
Member of classes: placePart [and indirectly also:] names
DTD file: teind2
Occurs within: placeName
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT country       - O  (%paraContent)                        >
<!ATTLIST country            %a.global
                             %a.placePart                          >
```

Discussed in: 20.2 ('Place Names') on p. 493

creation

<creation> contains information about the creation of a text.

Attributes: [None: global and inherited attributes only.]
Example:

```
<creation><date>Before 1987</date>
<creation><date value='1988-07-10'>10 July 1988</date>
```

The <creation> element may be used to record details of a text's creation, e.g. the date and place it was composed, if these are of interest; it should not be confused with the <publicationStmt> element, which records date and place of publication.
Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description: Character data and phrase-level elements.
Occurs within: profileDesc
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT creation      - O  (%phrase.seq;)                >
<!ATTLIST creation           %a.global                     >
```

Discussed in: 5.4.1 ('Creation') on p. 109; 5.4 ('The Profile Description') on p. 108

damage

<damage> contains an area of damage to the text witness.

Attributes:

type classifies the damage according to any convenient typology.
Data type: CDATA
Value: any phrase describing the damage, e.g. "faded", "overbound", "water", "charred with loss of paper".
Default: #IMPLIED

extent indicates approximately how much text is in the damaged area, in letters, minims, inches, or any appropriate unit, where this cannot be deduced from the contents of the tag. For example, the damage may span structural divisions in the text so that the tag must then be empty of content.
Data type: CDATA
Value: any measurement phrase, e.g. "25 letters", "2 x 3 inches".
Default: #IMPLIED

resp indicates the individual responsible for identifying the area of damage.
Data type: IDREF
Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).
Default: %INHERITED

hand In the case of damage (deliberate defacement, etc.) assignable to an identifiable hand, signifies the hand responsible for the damage.
Data type: IDREF
Value: must be one of the hand identifiers declared in the document header (see section 18.2.1 ('Document Hands') on p. 456).
Default: %INHERITED

agent In the case of damage resulting from an identifiable cause, signifies the causative agent.
Data type: CDATA
Value: any prose description of the agency of damage.
Default: #IMPLIED

degree Signifies the degree of damage according to a convenient scale. The <damage> tag with the degree attribute should only be used where the text may be read with some confidence; text supplied from other sources should be tagged as <supplied>.
Data type: CDATA
Value: an alphanumeric categorization of the degree of damage, as "40%".
Default: #IMPLIED

The <damage> tag with the degree attribute should only be used where the text may be read with confidence despite the damage. It is appropriate where it is desired to record the fact of damage, though this has not affected the readability of the text (as may be the case with weathered inscriptional materials). Where the damage has rendered the text more or less illegible either the <unclear> tag (for partial illegibility) or the <omit> tag (for complete illegibility, with no text supplied) should be used, with the information concerning the damage given in the attribute values of these tags. See section 18.2.4 ('The Use of the Gap, Del, Damage, Unclear and Supplied Tags in Combination') on p. 462 for discussion of the use of these tags in particular circumstances.

Since damage to text witnesses frequently makes them harder to read, the <damage> element will often contain an <unclear> element. If the damaged area is not continuous in the text (e.g. a stain on one side of a page), the <join> element may be used to indicate which <damage> and <unclear> elements are part of the same physical phenomenon.

The <damage>, <omit>, <del>, <unclear> and <supplied> elements may be closely allied in use. See section 18.2.4 ('The Use of the Gap, Del, Damage, Unclear and Supplied Tags in Combination') on p. 462 for discussion of which element is appropriate for which circumstance.

Part: additional tag set for text criticism
Member of classes:
DTD file: teitran2
Data description: May contain character data and phrase-level elements.
Occurs within: [none]
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT damage        - O  (%paraContent;)                  >

<!ATTLIST damage             %a.global
          type               CDATA               #IMPLIED
          extent             CDATA               #IMPLIED
          resp               IDREF               %INHERITED

          hand               IDREF               %INHERITED

          agent              CDATA               #IMPLIED
          degree             CDATA               #IMPLIED      >
```

Discussed in: 18.2.3 ('Damage, Illegibility, and Supplied Text') on p. 460

dataDesc

<dataDesc> specifies the legal content of the element being documented, noting any semantic or application-dependent constraints, as well as constraints enforced by the SGML content model.

Attributes: [None: global and inherited attributes only.]
Example:

```
<dataDesc>A declarative sentence, an adjectival phrase, or the word EMPTY.
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: A declarative sentence, an adjectival phrase, or the word EMPTY.
Occurs within: tagDoc
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT dataDesc      - O  (%phrase.seq)                    >
```

```
<!ATTLIST dataDesc            %a.global                       >
```

Discussed in: 27.1 ('The TagDoc Documentation Element') on p. 603

dataType

<dataType> specifies an SGML *declared value* for an attribute.

Attributes: [None: global and inherited attributes only.]

Example:

```
<dataType>CDATA
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: must be a legal SGML keyword, notation name or a name token group.
Occurs within: attDef
May contain: #PCDATA
Declaration:

```
<!ELEMENT dataType      - O  (#PCDATA)                        >
<!ATTLIST dataType            %a.global                       >
```

Discussed in: 27.1.1 ('The AttList Documentation Element') on p. 605

date

<date> contains a date in any format.

Attributes:

calendar indicates the system or calendar to which the date belongs.
Data type: CDATA
Value: Recommended values include: *Gregorian*, *Julian*, *Roman*, *Mosaic*, *Revolutionary*, *Islamic*.
Default: #IMPLIED
Example:

```
He was born on <date calendar=Gregorian>Feb. 22,
1732</date> (<date calendar=Julian
value='1732-2-22'>Feb. 11, 1731/32, O.S.</date>).
```

value gives the value of the date in some standard form, usually yyyy-mm-dd.
Data type: CDATA
Value: Any string representing a date in standard format; recommended form is "yyyy-mm-dd", as defined by ISO 8601: 1988, *Data elements and interchange formats — Information interchange — Representation of dates and times.*
Default: #IMPLIED
Example:

```
This list begins in the year 1632, more precisely
on Trinity Sunday, i.e. the Sunday after Pentecost,
in that year the <date calendar=Julian value=1632-6-6>
27th of May (old style)</date>.
```

For simple dates, the value should give the Gregorian date in the form (yyyy-mm-dd) specified by ISO 8601. More complicated dates or special applications may require another calendar or another form; these should be documented in the <stdVals> element in the TEI Header.

certainty indicates the degree of precision to be attributed to the date.
Data type: CDATA
Value: Any appropriate value, e.g. *ca.*, *approx*, *after*, *before*.
Default: #IMPLIED

Example:

```
<date value='1980-02' certainty=approx>early February 1980</date>
```

Example:

```
Given on the <date value='1977-06-12'>Twelfth Day of June
in the Year of Our Lord One Thousand Nine Hundred and
Seventy-seven of the Republic the Two Hundredth and first
and of the University the Eighty-Sixth.</date>
```

Example:

```
<date value='1990-09'>September 1990</date>
```

Part: additional tag set for common core features
Member of classes: data, date, terminologyInclusions [and indirectly also:] phrase
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell change channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateline dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur imprint interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publicationStmt publisher pubPlace purpose q quote rdg re recording ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term termEntry time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT date          - -  (%phrase.seq;)                    >

<!ATTLIST date                %a.global
          calendar            CDATA                #IMPLIED
          value               CDATA                #IMPLIED
          certainty           CDATA                #IMPLIED      >
```

Discussed in: 6.4.4 ('Dates and times') on p. 137; 5.2.4 ('Publication, Distribution, etc') on p. 86; 5.5 ('The Revision Description') on p. 112; 6.10.2.3 ('Imprint, Pagination, and Other Details') on p. 171; 13.2 ('Tags for Terminological Data') on p. 312; 23.2.3 ('The Setting Description') on p. 549; 20.4 ('Dates and Time') on p. 499

dateline

<dateline> contains a brief description of the place, date, time, etc. of production of a letter, newspaper story, or other work, prefixed or suffixed to it as a kind of heading or trailer.

Attributes: [None: global and inherited attributes only.]
Example:

```
<dateline>Walden, this 29. of August 1592</dateline>
```

Example:

```
...and his heart was going like mad and yes
I said yes I will Yes.</p>
<closer>
 <dateline>
   <name type=place>Trieste-Z&uuml;rich-Paris,</>
   <date>1914-1921</date>
 </dateline>
</closer>
</div>
</text>
```

Part: base tag set for common core features
Member of classes:
DTD file: teistr2
Occurs within: closer opener
May contain: #PCDATA address date name time
Declaration:

```
<!ELEMENT dateline      - O  (date | time | name | #PCDATA |
                              address)*                      >
<!ATTLIST dateline           %a.global                       >
```

Discussed in: 7.2.4 ('Content of Textual Divisions') on p. 194; 7.2.2 ('Openers and Closers') on p. 192

dateRange

<dateRange> (i.e. date range) contains two dates or another phrase delimiting a time period.

Attributes:

calendar indicates the system or calendar to which the date belongs.
Data type: CDATA
Value: Recommended values include: *Gregorian*, *Julian*, *Roman*, *Mosaic*, *Revolutionary*, *Islamic*.
Default: #IMPLIED

from indicates the starting point of the period in standard form.
Data type: CDATA
Value: any date in a standard form; recommended form is yyyy-mm-dd.
Default: #IMPLIED
The value should conform to the standard form declared in the <std.vals> element in the TEI header.

to indicates the ending point of the period in standard form.
Data type: CDATA
Value: any date in a standard form; recommended form is yyyy-mm-dd.
Default: #IMPLIED
The value should conform to the standard form declared in the <stdVals> element in the TEI header.

exact indicates the precision to be attached to either or both dates specified.
Data type: (TO | FROM | BOTH | NONE)
Sample values include:

to the to date is exact
from the from date is exact
both both dates are exact
none both dates are approximate or unspecified

Default: #IMPLIED
Example:

```
<dateRange from=1100 to=1120 exact=from>
     Early 12th century</dateRange>
```

date values should conform to the standard form declared in the <stdVals> element in the TEI header.

Example:

```
He edited (<dateRange from=1846 to=1848>1846-48</dateRange>)
the first collection of Hungarian folk poetry.
```

Part: additional tag set for common core features
Member of classes: data, date [and indirectly also:] phrase
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage

tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT dateRange     - O  (%phrase.seq;)                    >

<!ATTLIST dateRange          %a.global
          calendar           CDATA               #IMPLIED
          from               CDATA               #IMPLIED
          to                 CDATA               #IMPLIED
          exact              (to | from | both | none)
                                                 #IMPLIED       >
```

Discussed in: 6.4.4 ('Dates and times') on p. 137

<dateStruct> contains an internally structured representation of a date. dateStruct

Attributes:

calendar indicates the system or calendar to which the date belongs.
Data type: CDATA
Value: Recommended values include: *Gregorian*, *Julian*, *Roman*, *Mosaic*, *Revolutionary*, *Islamic*.
Default: #IMPLIED

exact indicates the degree of precision to be attributed to the date.
Data type: CDATA
Value: Any appropriate value, e.g. *ca.*, *approx*, *after*, *before*.
Default: #IMPLIED

Example:

```
<dateStruct value='1990-09'>
  <month>September</month>
  <year>1990</year>
</dateStruct>
```

Example:

```
Given on the
<dateStruct value='1977-06-12'>
    <day reg=12>  Twelfth</day> Day of
    <month reg=06>June</month>
    <year reg=1977 type=nominal>in the Year of Our Lord
    One Thousand Nine Hundred and Seventy-seven</year>
    <dateStruct>
      <offset>of</offset><calEvent>the Republic</calEvent>
      <distance>the Two Hundredth and first
    </dateStruct>and
    <dateStruct>
      <offset>of</offset><calEvent>the University</calEvent>
      <distance>the Eighty-Sixth.
    </dateStruct>
</dateStruct>
```

Part: additional tag set for common core features

Member of classes: data, date, temporalExpr, terminologyInclusions [and indirectly also:] phrase

DTD file: teind2

Data description: May contain character data and temporal expression elements.

Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange dateStruct def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality

figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term termEntry time timeRange timeStruct title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: #pcdata dateStruct day distance hour minute month occasion offset second timeStruct week year

Declaration:

```
<!ELEMENT dateStruct    - -  ((%m.temporalExpr; | #pcdata)+)    >

<!ATTLIST dateStruct         %a.global
                             %a.temporalExpr
          calendar           CDATA               #IMPLIED
          exact              CDATA               #IMPLIED       >
```

Discussed in: 6.4.4 ('Dates and times') on p. 137; 5.2.4 ('Publication, Distribution, etc') on p. 86; 5.5 ('The Revision Description') on p. 112; 6.10.2.3 ('Imprint, Pagination, and Other Details') on p. 171; 13.2 ('Tags for Terminological Data') on p. 312; 23.2.3 ('The Setting Description') on p. 549; 20.4 ('Dates and Time') on p. 499

day

<day> (i.e. day) the day component of a structured date.

Attributes: [None: global and inherited attributes only.]

Example:

```
<dateStruct value='14-05-1993'>
  <day type=name>Friday</day>,
  <day type=number>14</day>
  <month type=name>May</month>
  <year>1993</year></dateStruct>
```

Part: additional tag set for

Member of classes: temporalExpr

DTD file: teind2

Occurs within: dateStruct timeStruct

May contain: #PCDATA

Declaration:

```
<!ELEMENT day           - -  (#PCDATA)                      >

<!ATTLIST day                %a.global
                             %a.temporalExpr                >
```

Discussed in: 20.4 ('Dates and Time') on p. 499

def

<def> (i.e. definition) contains definition text in a dictionary entry.

Attributes: [None: global and inherited attributes only.]

Part: base tag set for printed dictionaries

Member of classes: dictionaries, dictionaryParts, dictionaryTopLevel

DTD file: teidict2

Data description: May contain character data and phrase-level elements.

Occurs within: entry etym hom re sense trans

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref

Declaration:

```
<!ELEMENT def           - O  (%paraContent;)                >

<!ATTLIST def                %a.global
                             %a.dictionaries                >
```

Discussed in: 12.3.3.1 ('Definitions') on p. 286

<default> specifies the default declared value for an attribute. default

Attributes: [None: global and inherited attributes only.]
Example:

```
<default>#IMPLIED
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: any legal SGML declared value or TEI-defined keyword
Occurs within: attDef
May contain: #PCDATA
Declaration:

```
<!ELEMENT default       - O  (#PCDATA)                    >
<!ATTLIST default            %a.global                    >
```

Discussed in: 27.1.1 ('The AttList Documentation Element') on p. 605

<del> (i.e. deletion) contains a letter, word or passage deleted, marked as deleted, or otherwise indicated as superfluous or spurious in the copy text by an author, scribe, annotator or corrector. del

Attributes:

rend (i.e. rendition) indicates how the deletion was indicated in the copy text.
Data type: CDATA
Sample values include:
subpunction dots below the line indicate matter to be deleted.
overstrike lines through the text indicated matter to be deleted.
erasure material to be deleted has been erased (but remains legible enough to transcribe).
bracketed brackets around the material indicate that it is spurious or superfluous.
Default: #IMPLIED

type classifies the type of deletion using any convenient typology.
Data type: CDATA
Value: any string identifying the class of deletion.
Default: #IMPLIED
No recommendation of any particular typology is made here; to record the manner in which the deletion is signaled, use rend, not type.

status may be used to indicate faulty deletions, e.g. strikeouts which include too much or too little text.
Data type: CDATA
Value: any description of flaws in the marking of a deletion, e.g. "excess left", "excess right", "short left", "short right".
Default: 'UNREMARKABLE'
Status information on each deletion is needed rather rarely except in critical editions from authorial manuscripts.

resp (i.e. responsible) signifies the editor or transcriber responsible for identifying the hand of the deletion.
Data type: IDREF
Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).
Default: %INHERITED

cert (i.e. certainty) signifies the degree of certainty ascribed to the identification of the hand of the deletion.
Data type: CDATA
Default: #IMPLIED

hand signifies the hand of the agent which made the deletion.
Data type: IDREF
Value: must be one of the hand identifiers declared in the document header (see section 18.2.1 ('Document Hands') on p. 456).
Default: %INHERITED

Example:

```
<l><del type=overtyped>Mein</del> Frisch
   <del type=overstrike>schwebt</del> weht der Wind
```

Cf. <omit>.

Degrees of uncertainty over what can still be read may be indicated by use of the <certainty> element (see 17 ('Certainty and Responsibility') on p. 435).

This element should be used for deletion of shorter sequences of text, typically single words or phrases. The <delSpan> element should be used for longer sequences of text, for those containing structural subdivisions, and for those containing overlapping additions and deletions.

The text deleted must be at least partially legible, in order for the encoder to be able to transcribe it. If it is not legible at all, the <del> tag should not be used. Rather, the <omit> tag should be employed to signal that text cannot be transcribed, with the value of the reason attribute giving the cause for the omission from the transcription as deletion. If it is not fully legible, the <unclear> element (available when using the additional tagset for transcription of primary sources) should be used to signal the areas of text which cannot be read with confidence. See further sections 18.1.7 ('Text Omitted from or Supplied in the Transcription') on p. 455 and, for the close association of the <del> tag with the <omit>, <damage>, <unclear> and <supplied> elements (the latter three tags available when using the additional tagset for transcription of primary sources), 18.2.4 ('The Use of the Gap, Del, Damage, Unclear and Supplied Tags in Combination') on p. 462.

The <del> tag should not be used for deletions made by editors or encoders. In these cases, either the <corr> tag or the <omit> tag should be used.

Part: additional tag set for common core features
Member of classes: edit [and indirectly also:] phrase
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT del           - -  (%phrase.seq;)                    >

<!ATTLIST del                 %a.analysis
                              %a.linking
                              %a.terminology
          id                  ID                  #IMPLIED
          n                   CDATA               #IMPLIED
          lang                IDREF               %INHERITED
```

```
rend            CDATA           #IMPLIED
type            CDATA           #IMPLIED
status          CDATA           'unremarkable'
resp            IDREF           %INHERITED

cert            CDATA           #IMPLIED
hand            IDREF           %INHERITED     >
```

Discussed in: 6.5.3 ('Additions, Deletions and Omissions') on p. 144

delSpan

<delSpan> (i.e. deleted span of text) marks the beginning of a longer sequence of text deleted, marked as deleted, or otherwise signaled as superfluous or spurious by an author, scribe, annotator, or corrector.

Attributes:

type classifies the deletion, using any convenient typology.
Data type: CDATA
Sample values include:
overstrike deletion indicated by line crossing out the text.
erasure deletion indicated by erasure of the text.
bracketed deletion indicated by brackets in the text or margin.
subpunction deletion indicated by dots beneath the letters deleted.
Default: #IMPLIED

resp (i.e. responsible) signifies the editor or transcriber responsible for identifying the hand of the deletion.
Data type: IDREF
Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).
Default: %INHERITED

cert (i.e. certainty) signifies the degree of certainty ascribed to the identification of the hand of the deletion.
Data type: CDATA
Default: #IMPLIED

hand signifies the hand of the agent which made the deletion.
Data type: IDREF
Value: must be one of the hand identifiers declared in the document header (see section 18.2.1 ('Document Hands') on p. 456).
Default: %INHERITED

to identifies the endpoint of the deleted passage, by giving the ID of an <anchor> or other element placed there.
Data type: IDREF
Value: any valid SGML identifier, used on a later element.
Default: #REQUIRED
Both the beginning and the end of the deleted material must be marked; the beginning by the <delSpan> element itself, the end by the to attribute. The element pointed at by to is understood to be included within the deleted passage.

status indicates whether the deletion is faulty, e.g. by including too much or too little text.
Data type: CDATA
Sample values include:
excess start some text at the beginning of the deletion is marked as deleted even though it clearly should not be deleted.
excess end some text at the end of the deletion is marked as deleted even though it clearly should not be deleted.
short start some text at the beginning of the deletion is not marked as deleted even though it clearly should be.
short end some text at the end of the deletion is not marked as deleted even though it clearly should be.

unremarkable the deletion is not faulty.

Default: 'UNREMARKABLE'

Marking a deletion as faulty is inescapably an interpretive act; the usual test applied in practice is the linguistic acceptability of the text with and without the letters or words in question.

Example:

```
<p>Paragraph partially deleted.  This is the undeleted
portion <delSpan resp=author to=a23>and this the deleted
portion of the paragraph.
<p>Paragraph deleted together with neighboring material.
<p>Second fully deleted paragraph.
<p>Paragraph partially deleted; in the middle of this
paragraph the deletion ends and the anchor point marks
the resumption <anchor id=a23> of the text. ...
```

Both the beginning and ending of the deleted sequence must be marked: the beginning by the <delSpan> element, the ending by the target of the to attribute.

The text deleted must be at least partially legible, in order for the encoder to be able to transcribe it. If it is not legible at all, the <delSpan> tag should not be used. Rather, the <omit> tag should be employed to signal that text cannot be transcribed, with the value of the reason attribute giving the cause for the omission from the transcription as deletion. If it is not fully legible, the <unclear> element should be used to signal the areas of text which cannot be read with confidence. See further sections 18.1.7 ('Text Omitted from or Supplied in the Transcription') on p. 455 and, for the close association of the <delSpan> tag with the <omit>, <damage>, <unclear> and <supplied> elements, 18.2.4 ('The Use of the Gap, Del, Damage, Unclear and Supplied Tags in Combination') on p. 462.

The <delSpan> tag should not be used for deletions made by editors or encoders. In these cases, either the <corr> tag or the <omit> tag should be used.

Part: additional tag set for text criticism

Member of classes:

DTD file: teitran2

Data description: May contain character data and phrase-level elements.

Occurs within: [none]

May contain: [none]

Declaration:

```
<!ELEMENT delSpan       - O  EMPTY                              >

<!ATTLIST delSpan            %a.global
          type               CDATA              #IMPLIED
          resp               IDREF              %INHERITED

          cert               CDATA              #IMPLIED
          hand               IDREF              %INHERITED

          to                 IDREF              #REQUIRED
          status             CDATA              'unremarkable' >
```

Discussed in: 18.1.4 ('Additions and Deletions') on p. 449

derivation

<derivation> describes the nature and extent of indebtedness or derivativeness of this text with respect to others.

Attributes:

type categorizes the derivation of the text.

Data type: CDATA

Sample values include:

original text is original

revision text is a revision of some other text

translation text is a translation of some other text

abridgment text is an abridged version of some other text

plagiarism text is plagiarized from some other text

traditional text has no obvious source but is one of a number derived from some common ancestor

Default: #IMPLIED

Example:

```
<derivation type=original>
```

For derivative texts, details of the ancestor may be included in the source description.

Part: auxiliary tag set for corpora and collections
Member of classes:
DTD file: teicorp2
Data description:
Occurs within: textDesc
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT derivation    - O  (%phrase.seq)                    >

<!ATTLIST derivation         %a.global
          type               CDATA               #IMPLIED      >
```

Discussed in: 23.2.1 ('The Text Description') on p. 541

<desc> (i.e. description) (in a writing system declaration) contains a description of a character or character form. — desc

Attributes: [None: global and inherited attributes only.]
Example:

```
<form string='a'>
  <desc>Latin lower-case letter A</desc>
<form>
<form string='' entityLoc='infinity'>
  <desc>author's private symbol for infinity:  a parabola open to
        the right, with a dot at its focal point.  In general, the
        character rests on the base line and rises to about the
        height of a typical ascender.</desc>
</form>
```

The <desc> element should usually contain the name of the character or character form; in some cases further information (e.g. the description of the character's shape) will be useful, as shown in the example.

Part: auxiliary tag set for writing system declarations
Member of classes:
DTD file: teiwsd2
Data description: May contain character data only.
Occurs within: attDef character classDoc entDoc form formula tagDoc valList
May contain: #PCDATA
Declaration:

```
<!ELEMENT desc          - O  (#PCDATA)                        >

<!ATTLIST desc               %a.global                        >
```

Discussed in: 25.4.2 ('Exceptions in the WSD') on p. 577

<desc> (i.e. description) contains a brief description of the purpose and application for an element, attribute, or attribute value. — desc

Attributes: [None: global and inherited attributes only.]
Example:

```
<desc>contains a brief description of the purpose and application for
an element, attribute, attribute value, class, or entity.
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: TEI convention requires that this be expressed as a finite clause, begining with an active verb.
Occurs within: attDef character classDoc entDoc form formula tagDoc valList
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT desc          - O  (%paraContent)                    >
<!ATTLIST desc               %a.global                         >
```

Discussed in: 27.1 ('The TagDoc Documentation Element') on p. 603; 27.1.1 ('The AttList Documentation Element') on p. 605; 27.2 ('Element Classes') on p. 606; 27.3 ('Entity Documentation ') on p. 607

descrip

<descrip> (i.e. description) within a <termEntry> element, contains a definition, context or explanation used to explain or define the concept represented by a <term> or an <otherForm>.

Attributes:

type classifies the description using some convenient typology, preferably the dictionary of data element types specified in ISO WD 12 620.
Data type: CDATA
Sample values include:

definition The description provides all the information needed to differentiate one concept from all other related concepts in the given domain.

Default: #IMPLIED
A much fuller list of values for the type attribute may be generated from the dictionary of data element types under preparation as ISO TC 37/SC 3/WD 12 620, Computational Aids in Terminology. See ISO 12 620 for fuller details.

(optional)

Part: base tag set for terminological data
Member of classes: terminologyMisc
DTD file: teite2n teite2f
Data description: May contain character data and phrase-level elements.
Occurs within: ofig termEntry tig
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT descrip       - O  (%paraContent;)                   >
<!ATTLIST descrip            %a.global
          type               CDATA              #IMPLIED       >
```

Discussed in: 13.4.2 ('DTD Fragment for Flat Style') on p. 322; 13.4.1 ('DTD Fragment for Nested Style') on p. 321; 13.2 ('Tags for Terminological Data') on p. 312

dft

<dft> (i.e. Default value) provides default value for a feature.

Attributes: [None: global and inherited attributes only.]
Example:

```
<f name=gender><dft></f>
```

Part: additional tag set for feature structures
Member of classes: singleVal [and indirectly also:] featureVal
DTD file: teifs2
Data description: Empty element.
Occurs within: f fvLib if vAlt vDefault vRange
May contain: [none]
Declaration:

```
<!ELEMENT dft           - O  EMPTY                          >
<!ATTLIST dft                %a.global                      >
```

Discussed in: 16.8 ('Boolean, Default and Uncertain Values') on p. 417

direction

<direction> specifies one or more conventional directions in which a language is written using a given script.

Attributes:

chars (i.e. characters) indicates the order in which characters within a line are conventionally presented in this writing system
Data type: CDATA
Sample values include:
LR left to right
RL right to left
TB top to bottom
BT bottom to top
Default: #REQUIRED
If the characters follow each other in some order other than top-to-bottom, bottom-to-top, left-to-right, or right-to-left, then some value other than those indicated should be used.

lines indicates the order in which lines conventionally follow each other in this writing system.
Data type: CDATA
Sample values include:
TB top to bottom
BT bottom to top
LR left to right
RL right to left
Default: #REQUIRED
If the lines follow each other in some order other than top-to-bottom, bottom-to-top, left-to-right, or right-to-left, then some value other than those indicated should be used.

Example:

```
<writingSystemDeclaration lang=eng id='deu'
 name='-//TEI P2: 1993//WSD Modern German//en' date='1993-06-01'>
 <language iso639='deu'>New High German</language>
 <script>Latin script with diacritics.</script>
 <direction chars='LR' lines='TB'>
 <!-- ... -->
</writingSystemDeclaration>
```

If more than one direction is specified, it means that the script in question may be written in any of the directions named. In some cases, the directions may be mixed within a single document, in others not; no information is given on this point by the writing system declaration, as it is not usefully formalizable. Experts on the script must be consulted for details. The following example shows one way to declare that a script may be written in any horizontal or vertical direction.

Example:

```
<writingSystemDeclaration lang=eng id=jpn
 name='-//TEI P2: 1993//WSD JIS 0208//en' date='1993-06-01'>
 <language iso639='jpn'>Modern Japanese</language>
```

```
<script>normal Japanese writing, with mixture of hiragana,
        katakana, and kanji.</script>
<direction chars='LR' lines='TB'>
<direction chars='RL' lines='TB'>
<direction chars='LR' lines='BT'>
<direction chars='RL' lines='BT'>
<direction chars='TB' lines='LR'>
<direction chars='BT' lines='LR'>
<direction chars='TB' lines='RL'>
<direction chars='BT' lines='RL'>
<!-- Japanese can, in effect, be written in any direction. -->
<!-- Individual documents may, but need not, mix the       -->
<!-- various directions.                                   -->
<!-- ... -->
</writingSystemDeclaration>
```

The following example shows another way to specify that a script may be written in any horizontal or vertical direction.

Example:

```
<writingSystemDeclaration lang=eng id=jpn
 name='//TEI P2: 1993//WSD JIS 0208//en' date='1993-06-01'>
 <language iso639='jpn'>Modern Japanese</language>
 <script>normal Japanese, with mixture of hiragana, katakana,
        and kanji.</script>
 <direction chars='LR RL' lines='TB BT'>
 <direction chars='TB BT' lines='LR RL'>
 <!-- Japanese can, in effect, be written in any direction. -->
 <!-- Individual documents may, but need not, mix the       -->
 <!-- various directions.                                   -->
 <!-- ... -->
</writingSystemDeclaration>
```

In some cases, the lines and chars attributes may need to take special values. Some scripts are written in "boustrophedon" (turning back and forth): i.e. one line is written left to right, the next right to left, and so on. Such a writing system might have its direction declared as shown:

Example:

```
<direction chars='boustrophedon:  LR, then RL, then LR, etc.'
           lines='TB'>
```

This element describes conventional presentation; all scripts are subject to unusual treatment in special circumstances, and such unusual directions need not be described here. The treatment of numerals in Latin, Hebrew, and Arabic script, being well understood, need not be documented separately here.[1]

The <direction> element is informational only, not normative. It is intended to alert those responsible for implementing support for a given writing system to an essential fact of how it is written. If only a single direction is specified, it is safe to infer that the script may legitimately be presented in that direction. If multiple directions are specified, it may be legitimate to present text in any one of them, or it may be necessary to support multiple directions in display of a single document. If the latter, then changes of direction should be given using the global rend attribute.

If no direction element is given, the only safe assumption is that any direction is possible and experts must be consulted before attempting to implement support for the writing system.

Part: auxiliary tag set for writing system declarations
Member of classes:
DTD file: teiwsd2
Data description: Empty.
Occurs within: writingSystemDeclaration

[1] Although the scripts run in opposite directions, they write numbers in the same direction; the usual view is that the numbers in Hebrew and Arabic run left to right, like those in Latin script, but it is also possible to claim that the numbers in Latin scripts run right to left, like those in Arabic and Hebrew. There is no single satisfactory answer to this question.

May contain: [none]
Declaration:

```
<!ELEMENT direction     - O  EMPTY                              >

<!ATTLIST direction           %a.global
          chars               CDATA                #REQUIRED
          lines               CDATA                #REQUIRED   >
```

Discussed in: 25.3 ('Describing the Writing System') on p. 574; 25.1 ('Overall Structure of Writing System Declaration') on p. 571

<distance> (i.e. distance) that part of a relative temporal or spatial expression which indicates the distance between the place or time denoted by it and the place or time referred to within it. distance

Attributes:

exact indicates the degree of accuracy associated with the distance.
Data type: (Y | N | U)
Sample values include:

Y The distance is exact.

N The distance is approximate.

U Accuracy unavailable or unknown.

Default: U

reg provides a normalized evaluation of the expression of relative distance.
Data type: CDATA
Value:
Default: #IMPLIED

Example:

```
<relPlace>
<distance>two miles</distance>
<offset>north east of</offset>
<placeName> Manchester</placeName>
</relPlace>

<dateStruct>
  <distance exact=N reg='14 days'>
     a fortnight</distance>
  <offSet>after</offset>
  <calEvent>Michaelmas</calEvent>
</dateStruct>

<timeStruct>
  <distance>20 minutes</distance>
  <offset>before</offset>
  <calEvent>noon</calEvent>.
</timeStruct>
```

This is a specialized form of measure, used only within relative temporal and spatial expressions.
Part: auxiliary tag set for names and dates
Member of classes: placePart, temporalExpr [and indirectly also:] names
DTD file: teind2
Occurs within: dateStruct placeName timeStruct
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT distance      - -  (%phrase.seq)                      >
```

```
<!ATTLIST distance          %a.global
          key               CDATA               #IMPLIED
          value             CDATA               #IMPLIED
          type              CDATA               #IMPLIED
          full              (yes | abb | init)  yes
          exact             (Y | N | U)         U
          reg               CDATA               #IMPLIED      >
```

Discussed in: 20.2 ('Place Names') on p. 493

distinct

<distinct> (i.e. distinct) identifies any word or phrase which is regarded as linguistically distinct, for example as archaic, technical, dialectal, non-preferred, etc., or as forming part of a sublanguage.

Attributes:

type specifies the sublanguage or register to which the word or phrase is being assigned
Data type: CDATA
Value: a semi-open user-defined list
Default: #IMPLIED

time specifies how the phrase is distinct diachronically
Data type: CDATA
Value: a semi-open user-defined list
Default: #IMPLIED

space specifies how the phrase is distinct diatopically
Data type: CDATA
Value: a semi-open user-defined list
Default: #IMPLIED

social specifies how the phrase is distinct diastatically
Data type: CDATA
Value: a semi-open user-defined list
Default: #IMPLIED

Example:

```
Next morning a boy in that dormitory confided to his
bosom friend, a <distinct type=ps_slang>fag</distinct> of
Macrea's, that there was trouble in their midst which
King <distinct type=archaic>would fain</distinct> keep
secret.
```

Example:

```
<!-- need another example -->
```

Part: base tag set for common core features
Member of classes: hqphrase [and indirectly also:] phrase
DTD file: teicore2
Data description:
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT distinct      - -  (%phrase.seq;)                    >

<!ATTLIST distinct           %a.global
          type               CDATA               #IMPLIED
          time               CDATA               #IMPLIED
          space              CDATA               #IMPLIED
          social             CDATA               #IMPLIED      >
```

Discussed in: 6.3.2.3 ('Other Linguistically Distinct Material') on p. 126; 6.3.2 ('Emphasis, Foreign Words, and Unusual Language') on p. 124

<distributor> supplies the name of a person or other agency responsible for the distribution of a text. distributor

Attributes: [None: global and inherited attributes only.]
Example:

```
<distributor>Oxford Text Archive
<distributor>Redwood and Burn Ltd
```

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:
Occurs within: publicationStmt
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT distributor   - O  (%phrase.seq;)                    >

<!ATTLIST distributor        %a.global                         >
```

Discussed in: 5.2.4 ('Publication, Distribution, etc') on p. 86

(i.e. text division) contains a subdivision of the front, body, or back of a text. div

Attributes: [None: global and inherited attributes only.]
Example:

```
<body>
<div type=part><head>Fallacies of Authority</head>
<epigraph>The subject of which is Authority in
various shapes, and the object, to repress all
exercise of the reasoning faculty.</epigraph>
  <div type=chapter n=1><head>The Nature of Authority</head>
  <p>With reference to any proposed measures
  having for their object the greatest happiness
  of the greatest number....
     <div type=section n='1.1'><head>Analysis of Authority
      <p>What on any given occasion is the legitimate
      weight or influence to be attached to authority
      <!-- ... -->
     </div>
     <div type=section n='1.2'><head>Appeal to Authority,
        in What Cases Fallacious.
      <p>Reference to authority is open to the charge of
      fallacy when...
      <!-- ... -->
     </div>
     <!-- other sections here -->
   </div>
   <div type=chapter n=2>
   <!-- other chapters here -->
```

```
    </div>
  </div>
  <!-- other parts here -->
  </body>
```

Part: base tag set for common core features
Member of classes: declaring, divn
DTD file: teistr2
Data description: any sequence of low-level structural elements, possibly grouped into lower subdivisions.
Occurs within: back body div front
May contain: argument bibl biblFull biblStruct byline camera caption castList cit closer div docAuthor docDate entry entryFree epigraph event eTree graph head kinesic l label lg lg1 list listBibl move note opener p pause q quote salute shift signed sound sp stage superentry tech termEntry trailer tree u view vocal writing
Declaration:

```
<!ELEMENT div           - O  ((%m.divtop;)*, (div+ |
                              ((%component;)+, div*)),
                              (%m.divbot;)*)                    >

<!ATTLIST div                %a.global
                             %a.declaring
                             %a.divn                            >
```

Discussed in: 7.1 ('Divisions of the Body') on p. 185; 13.4 ('Overall Structure of Terminological Documents') on p. 319; 7.1.1 ('Un-numbered Divisions') on p. 185

divGen

<divGen> (i.e. automatically generated text division) indicates the location at which a textual division generated automatically by a text-processing application is to appear.

Attributes:

type specifies what type of generated text division (e.g. index, table of contents, etc.) is to appear.
Data type: CDATA
Sample values include:

index an index is to be generated and inserted at this point.
toc a table of contents
figlist a list of figures
tablist a list of tables

Default: #IMPLIED
Valid values are application-dependent; those shown are of obvious utility in document production, but are by no means exhaustive.

One use for this element is to allow document preparation software to generate an index and insert it in the appropriate place in the output. The example below assumes that the index attribute on the <index> element has been used to specify index entries for multiple indices, 1 and 2.

Example:

```
<back>
<div1><head>Bibliography</head>
<listBibl> ... </listBibl>
<divGen type='index 1' n='Index Nominum'>
<divGen type='index 2' n='Index Rerum'>
</back>
```

Another use for <divGen> is to specify the location of an automatically produced table of contents:

Example:

```
<front>
<titlePage> ... </titlePage>
<divGen type='toc'>
```

```
<div><head>Preface</head>
<p> ...
</front>
```

This element is intended primarily for use in document production or manipulation, rather than in the transcription of pre-existing materials; it makes it easier to specify the location of indices, tables of contents, etc., to be generated by text preparation or word processing software. The n attribute should be used to give a title for the text division being generated.
Part: additional tag set for common core features
Member of classes:
DTD file: teicore2
Data description: Empty.
Occurs within: [none]
May contain: [none]
Declaration:

```
<!ELEMENT divGen        - O  EMPTY                              >

<!ATTLIST divGen             %a.global
          type               CDATA               #IMPLIED      >
```

Discussed in: 6.8.2 ('Index Entries') on p. 154

<div0> (i.e. level-0 text division) contains the largest possible subdivision of the body of a text. div0

Attributes: [None: global and inherited attributes only.]
Example:

```
<div0 n=I id='LEVI' type=part>
   <head>Part I: Of Man </head>
   <div1 n='1' id='LEVI1' type=chapter>
     <head>Chap. I. Of Sense </head>
     <p>Concerning the Thoughts of man...
   </div1>
    <!-- further chapters here -->
</div0>
<div0 n=II id=LEVII type=part>
   <head>Part II: Of Common-Wealth</head>
   <!-- ... -->
</div0>
```

Part: base tag set for common core features
Member of classes: declaring, divn
DTD file: teistr2
Data description: any sequence of low-level structural elements, possibly grouped into lower subdivisions.
Occurs within: body
May contain: argument bibl biblFull biblStruct byline camera caption castList cit closer div1 docAuthor docDate entry entryFree epigraph event eTree graph head kinesic l label lg lg1 list listBibl move note opener p pause q quote salute shift signed sound sp stage superentry tech termEntry trailer tree u view vocal writing
Declaration:

```
<!ELEMENT div0          - O  ((%m.divtop;)*, ( div1+ | (
                             (%component;)+, div1*)),
                             (%m.divbot;)*)                  >

<!ATTLIST div0               %a.global
                             %a.declaring
                             %a.divn                         >
```

Discussed in: 7.1.2 ('Numbered Divisions') on p. 186; 13.4 ('Overall Structure of Terminological Documents') on p. 319

<div1> (i.e. level-1 text division) contains a first-level subdivision of the front, body, or back of a text (the largest, if <div0> is not used, the second largest if it is). div1

Attributes: [None: global and inherited attributes only.]
Example:

```
<div1 n=I id='LEVI' type=part>
   <head>Part I: Of Man </head>
   <div2 n='1' id='LEVI1' type=chapter>
     <head>Chap. I. Of Sense </head>
     <p>Concerning the Thoughts of man...
   </div2>
    <!-- further chapters here -->
</div1>
<div1 n=II id=LEVII type=part>
   <head>Part II: Of Common-Wealth</head>
   <!-- ... -->
</div1>
```

Part: base tag set for common core features
Member of classes: declaring, divn
DTD file: teistr2
Data description: any sequence of low-level structural elements, possibly grouped into lower subdivisions.
Occurs within: back body div0 front
May contain: argument bibl biblFull biblStruct byline camera caption castList cit closer div2 docAuthor docDate entry entryFree epigraph event eTree graph head kinesic l label lg lg1 list listBibl move note opener p pause q quote salute shift signed sound sp stage superentry tech termEntry trailer tree u view vocal writing
Declaration:

```
<!ELEMENT div1          - O  ((%m.divtop;)*, ( div2+ |
                             ((%component;)+, div2*)),
                             (%m.divbot;)*)                    >

<!ATTLIST div1               %a.global
                             %a.declaring
                             %a.divn                           >
```

Discussed in: 7.1.2 ('Numbered Divisions') on p. 186; 13.4 ('Overall Structure of Terminological Documents') on p. 319

div2

<div2> (i.e. level-2 text division) contains a second-level subdivision of the front, body, or back of a text.

Attributes: [None: global and inherited attributes only.]
Example:

```
<div1 n=2 type=part><head>The Second Partition:
              The Cure of Melancholy</head>
  <div2 n=2.1 type=section>
    <div3 n=2.1.1 type=member>
      <div4 n=2.1.1.1 type=subsection>
        <head>Unlawful Cures rejected.</head>
        <p>Inveterate melancholy, howsoever it may seem to
         be a continuate, inexorable disease, hard to be
         cured, accompanying them to their graves most part
         (as <ref target=A>Montanus</ref> observes), yet many
         times it may be helped...
      <!-- ... -->
  </div2>
  <div2 n=2.2>
   <div3 n=2.2.1><head>Sect. II. Memb. I</head>
      <!-- ... -->
  </div2>
  <div2 n=2.3>
   <div3 n=2.3.1><head>Sect. III. Memb. I</head>
     <!-- ... -->
  </div2>
```

Part: base tag set for common core features
Member of classes: declaring, divn
DTD file: teistr2
Data description: any sequence of low-level structural elements, possibly grouped into lower subdivisions.
Occurs within: div1
May contain: argument bibl biblFull biblStruct byline camera caption castList cit closer div3 docAuthor docDate entry entryFree epigraph event eTree graph head kinesic l label lg lg1 list listBibl move note opener p pause q quote salute shift signed sound sp stage superentry tech termEntry trailer tree u view vocal writing
Declaration:

```
<!ELEMENT div2          - O  ((%m.divtop;)*, ( div3+ |
                             ((%component;)+, div3*)),
                             (%m.divbot;)*)                    >

<!ATTLIST div2               %a.global
                             %a.declaring
                             %a.divn                           >
```

Discussed in: 7.1.2 ('Numbered Divisions') on p. 186; 13.4 ('Overall Structure of Terminological Documents') on p. 319

<div3> (i.e. level-3 text division) contains a third-level subdivision of the front, body, or back of a text. — div3

Attributes: [None: global and inherited attributes only.]
Example:

```
<div2 n=2.2 type=section>
 <div3 n=2.2.1 type=member><head>Sect. II. Memb. I</head>
      <!-- ... -->
 <div3 n=2.2.2 type=member><head>Memb. II
      Retention and Evacuation rectified.</head>
     <!-- ... -->
 <div3 n=2.2.3 type=member><head>Memb. III
      Ayr rectified. With a digression of the Ayr.</head>
     <!-- ... -->
 </div2>
```

Part: base tag set for common core features
Member of classes: declaring, divn
DTD file: teistr2
Data description: any sequence of low-level structural elements, possibly grouped into lower subdivisions.
Occurs within: div2
May contain: argument bibl biblFull biblStruct byline camera caption castList cit closer div4 docAuthor docDate entry entryFree epigraph event eTree graph head kinesic l label lg lg1 list listBibl move note opener p pause q quote salute shift signed sound sp stage superentry tech termEntry trailer tree u view vocal writing
Declaration:

```
<!ELEMENT div3          - O  ((%m.divtop;)*, ( div4+ |
                             ((%component;)+, div4*)),
                             (%m.divbot;)*)                    >

<!ATTLIST div3               %a.global
                             %a.declaring
                             %a.divn                           >
```

Discussed in: 7.1.2 ('Numbered Divisions') on p. 186; 13.4 ('Overall Structure of Terminological Documents') on p. 319

<div4> (i.e. level-4 text division) contains a fourth-level subdivision of the front, body, or back of a text. — div4

Attributes: [None: global and inherited attributes only.]
Example:

```
<div3 n=2.2.1><head>Sect. II. Memb. I</head>
 <div4 n=2.2.1.1 type=subsection>
  <head>Subsect I. -- Dyet rectified in substance.</head>
  <p>Diet, <term lang=GRK>diaitotiku</term>,
  <term lang=LAT>victus</term>  or living
  <!-- ... -->
 </div4>
 <div4 n=2.2.2.1 type=subsection>
  <head>Subsect II. -- Dyet rectified in quantity.</head>
  <p>Man alone, saith Cardan, eates and drinks without
  appetite, and useth all his pleasures without necessity
  <!-- ... -->
 </div4>
</div3>
```

Part: base tag set for common core features
Member of classes: declaring, divn
DTD file: teistr2
Data description: any sequence of low-level structural elements, possibly grouped into lower subdivisions.
Occurs within: div3
May contain: argument bibl biblFull biblStruct byline camera caption castList cit closer div5 docAuthor docDate entry entryFree epigraph event eTree graph head kinesic l label lg lg1 list listBibl move note opener p pause q quote salute shift signed sound sp stage superentry tech termEntry trailer tree u view vocal writing
Declaration:

```
<!ELEMENT div4          - O  ((%m.divtop;)*, ( div5+ |
                             ((%component;)+, div5*)),
                             (%m.divbot;)*)                    >

<!ATTLIST div4               %a.global
                             %a.declaring
                             %a.divn                           >
```

Discussed in: 7.1.2 ('Numbered Divisions') on p. 186; 13.4 ('Overall Structure of Terminological Documents') on p. 319

div5

<div5> (i.e. level-5 text division) contains a fifth-level subdivision of the front, body, or back of a text.
Attributes: [None: global and inherited attributes only.]
Example:

```
<!-- see examples for higher level divisions -->
```

Part: base tag set for common core features
Member of classes: declaring, divn
DTD file: teistr2
Data description: any sequence of low-level structural elements, possibly grouped into lower subdivisions.
Occurs within: div4
May contain: argument bibl biblFull biblStruct byline camera caption castList cit closer div6 docAuthor docDate entry entryFree epigraph event eTree graph head kinesic l label lg lg1 list listBibl move note opener p pause q quote salute shift signed sound sp stage superentry tech termEntry trailer tree u view vocal writing
Declaration:

```
<!ELEMENT div5          - O  ((%m.divtop;)*, ( div6+ |
                             ((%component;)+, div6*)),
                             (%m.divbot;)*)                    >

<!ATTLIST div5               %a.global
                             %a.declaring
                             %a.divn                           >
```

Discussed in: 7.1.2 ('Numbered Divisions') on p. 186; 13.4 ('Overall Structure of Terminological Documents') on p. 319

div6

<div6> (i.e. level-6 text division) contains a sixth-level subdivision of the front, body, or back of a text.

Attributes: [None: global and inherited attributes only.]

Example:

```
<!-- see examples for higher level divisions -->
```

Part: base tag set for common core features

Member of classes: declaring, divn

DTD file: teistr2

Data description: any sequence of low-level structural elements, possibly grouped into lower subdivisions.

Occurs within: div5

May contain: argument bibl biblFull biblStruct byline camera caption castList cit closer div7 docAuthor docDate entry entryFree epigraph event eTree graph head kinesic l label lg lg1 list listBibl move note opener p pause q quote salute shift signed sound sp stage superentry tech termEntry trailer tree u view vocal writing

Declaration:

```
<!ELEMENT div6          - O  ((%m.divtop;)*, (div7+ |
                              ((%component;)+, div7*)),
                              (%m.divbot;)*)                    >

<!ATTLIST div6                %a.global
                              %a.declaring
                              %a.divn                           >
```

Discussed in: 7.1.2 ('Numbered Divisions') on p. 186; 13.4 ('Overall Structure of Terminological Documents') on p. 319

div7

<div7> (i.e. level-7 text division) contains the smallest possible subdivision of the front, body or back of a text, larger than a paragraph.

Attributes: [None: global and inherited attributes only.]

Example:

```
<!-- see examples for higher level divisions -->
```

Part: base tag set for common core features

Member of classes: declaring, divn

DTD file: teistr2

Data description: any sequence of low-level structural elements, possibly grouped into lower subdivisions.

Occurs within: div6

May contain: argument bibl biblFull biblStruct byline camera caption castList cit closer docAuthor docDate entry entryFree epigraph event eTree graph head kinesic l label lg lg1 list listBibl move note opener p pause q quote salute shift signed sound sp stage superentry tech termEntry trailer tree u view vocal writing

Declaration:

```
<!ELEMENT div7          - O  ((%m.divtop;)*, (%component;)+,
                              (%m.divbot;)*)                    >

<!ATTLIST div7                %a.global
                              %a.declaring
                              %a.divn                           >
```

Discussed in: 7.1.2 ('Numbered Divisions') on p. 186; 13.4 ('Overall Structure of Terminological Documents') on p. 319

docAuthor

<docAuthor> (i.e. document author) contains the name of the author of the document, as given on the title page (often but not always contained in a <byline>).

Attributes: [None: global and inherited attributes only.]

The document author's name often occurs within a byline, but the <docAuthor> element may be used whether the <byline> element is used or not.

Part: base tag set for common core features
Member of classes: divtop, tpParts
DTD file: teifron2
Data description: May contain character data and phrase-level elements.
Occurs within: back body byline castList div div0 div1 div2 div3 div4 div5 div6 div7 epilogue front group lg performance prologue titlePage
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT docAuthor      - O  (%phrase.seq;)                    >

<!ATTLIST docAuthor           %a.global                         >
```

Discussed in: 7.5 ('Title Pages') on p. 203

docDate

<docDate> (i.e. document date) contains the date of a document, as given (usually) on a title page.

Attributes:

value gives the value of the date in a standard form.

Data type: %ISO-DATE

Value: Any string representing a date in standard format; recommended form is "yyyy-mm-dd".

Default: #IMPLIED

Example:

```
This list begins in the year 1632, more precisely
on Trinity Sunday, i.e. the Sunday after Pentecost,
in that year the <date type=Julian value=1632-6-6>
27th of May (old style)</date>.
```

For simple dates, the value should give the Gregorian date in the form (yyyy-mm-dd) specified by ISO 8601. More complicated dates or special applications may require another calendar or another form; these should be documented in the <std.vals> element in the TEI Header.

Example:

```
<docImprint>Oxford, Clarendon Press, <docDate>1987</></docImprint>
```

Cf. the general <date> element in the core tag set. This specialized element is provided for convenience in marking and processing the date of the documents, since it is likely to require specialized handling for many applications.

Part: base tag set for common core features
Member of classes: divtop, tpParts
DTD file: teifron2
Data description:
Occurs within: back body castList div div0 div1 div2 div3 div4 div5 div6 div7 docImprint epilogue front group lg performance prologue titlePage
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT docDate        - O  (%phrase.seq;)                    >

<!ATTLIST docDate             %a.global
          value               %ISO-date           #IMPLIED     >
```

Discussed in: 7.5 ('Title Pages') on p. 203

docEdition

<docEdition> (i.e. document edition) contains an edition statement as presented on a title page of a document.

Attributes: [None: global and inherited attributes only.]
Example:

```
<docEdition>The Third edition Corrected</>
```

Cf. the <edition> element of bibliographic citation. As usual, the shorter name has been given to the more frequent element.
Part: base tag set for common core features
Member of classes: tpParts
DTD file: teifron2
Data description:
Occurs within: titlePage
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT docEdition    - O  (%paraContent;)                    >

<!ATTLIST docEdition         %a.global                          >
```

Discussed in: 7.5 ('Title Pages') on p. 203

docImprint

<docImprint> (i.e. document imprint) contains the imprint statement (place and date of publication, publisher name), as given (usually) at the foot of a title page.

Attributes: [None: global and inherited attributes only.]
Example:

```
<docImprint>Oxford, Clarendon Press, 1987</docImprint>
```

Imprints may be somewhat more complex:
Example:

```
<docImprint><pubPlace>London</pubPlace>
Printed for <name>E. Nutt</>,
at <pubPlace>Royal Exchange</>;
<name>J. Roberts</> in <pubPlace>wick-Lane</>;
<name>A. Dodd</> without <pubPlace>Temple-Bar</>;
and <name>J. Graves</> in <pubPlace>St. James's-street.</>
<date>1722.</>
</docImprint>
```

Cf. the <imprint> element of bibliographic citations. As with title, author, and editions, the shorter name is reserved for the element likely to be used more often.
Part: base tag set for common core features
Member of classes: tpParts
DTD file: teifron2
Data description:
Occurs within: titlePage
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct docDate emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr publisher pubPlace pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT docImprint    - O  (%phrase.seq | pubPlace | docDate
                              | publisher)*                     >

<!ATTLIST docImprint         %a.global                          >
```

Discussed in: 7.5 ('Title Pages') on p. 203

docTitle

<docTitle> (i.e. document title) contains the title of a document, including all its constituents, as given on a title page.

Attributes: [None: global and inherited attributes only.]
Example:

```
<docTitle>
<titlePart type=main>The DUNCIAD,
VARIOURVM.
<titlePart type=sub>WITH THE
PROLEGOMENA of SCRIBLERUS.
</docTitle>
```

Part: base tag set for common core features
Member of classes: tpParts
DTD file: teifron2
Data description:
Occurs within: titlePage
May contain: titlePart
Declaration:

```
<!ELEMENT docTitle      - O  (titlePart+)                    >

<!ATTLIST docTitle           %a.global                       >
```

Discussed in: 7.5 ('Title Pages') on p. 203

domain

<domain> (i.e. Domain of use) describes the most important social context in which the text was realized or for which it is intended, for example private vs. public, education, religion, etc.

Attributes:

type categorizes the domain of use.
Data type: CDATA
Sample values include:

art art and entertainment
domestic domestic and private
religious religious and ceremonial
business business and work place
education education
govt government and law
public other forms of public context

Default: #IMPLIED

Example:

```
<domain type=dom>
<domain type=rel>religious broadcast</domain>
```

The list presented here is primarily for illustrative purposes.
Part: auxiliary tag set for corpora and collections
Member of classes:
DTD file: teicorp2
Data description: Usually empty, unless some further clarification of the type attribute is needed, in which case it may contain running prose.
Occurs within: textDesc
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT domain        - O  (%phrase.seq)                   >

<!ATTLIST domain             %a.global
          type               CDATA              #IMPLIED     >
```

Discussed in: 23.2.1 ('The Text Description') on p. 541

<edition> (i.e. Edition) describes the particularities of one edition of a text.

Attributes: [None: global and inherited attributes only.]
Example:

```
<edition>First edition <date>Oct 1990</date>
<edition n='S2'>Students' edition
```

Part: auxiliary tag set for TEI headers
Member of classes: biblPart
DTD file: teihdr2
Data description:
Occurs within: bibl editionStmt monogr
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT edition       - O  (%phrase.seq;)                    >

<!ATTLIST edition            %a.global                         >
```

Discussed in: 5.2.2 ('The Edition Statement') on p. 84

<editionStmt> (i.e. edition statement) groups information relating to one edition of a text. editionStmt

Attributes: [None: global and inherited attributes only.]
Example:

```
<editionStmt>
  <edition n=S2>Students' edition</edition>
  <respStmt>
    <resp>Adapted by </resp><name>Elizabeth Kirk</name>
  </respStmt>
</editionStmt>
```

Example:

```
<editionStmt><p>First edition, <date>Michaelmas Term, 1991.</date>
</editionStmt>
```

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:
Occurs within: biblFull fileDesc
May contain: edition p respStmt
Declaration:

```
<!ELEMENT editionStmt   - O  ( (edition, respStmt*) | p+ )     >

<!ATTLIST editionStmt        %a.global                         >
```

Discussed in: 5.2.2 ('The Edition Statement') on p. 84; 5.2 ('The File Description') on p. 80

<editor> (i.e. editor) secondary *statement of responsibility* for a bibliographic item, for example the name of an individual, institution or organization, (or of several such) acting as editor, compiler, translator, etc. editor

Attributes:

role specifies the nature of the intellectual responsibility
Data type: CDATA
Value: semi-open list (examples might include: translator, editor, compiler, illustrator, etc.)
Default: EDITOR

Example:

```
<editor>Eric Johnson
<editor role=illustrator>John Tenniel
```

Particularly where cataloguing is likely to be based on the content of the header, it is advisable to use generally recognized authority lists for the exact form of personal names.
Part: base tag set for common core features
Member of classes: biblPart
DTD file: teicore2
Data description: A consistent format should be adopted
Occurs within: analytic bibl monogr series titleStmt
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT editor        - O  (%phrase.seq)                    >

<!ATTLIST editor             %a.global
          role               CDATA               editor      >
```

Discussed in: 6.10.2.2 ('Authors, Titles, and Editors') on p. 168

editorialDecl

<editorialDecl> (i.e. editorial practice declaration) provides details of editorial principles and practices applied during the encoding of a text.

Attributes: [None: global and inherited attributes only.]
Example:

```
<editorialDecl id=E2>
  <normalization source=W9>
  <p>All words converted to Modern American spelling using
    Websters 9th Collegiate dictionary
  <quotation marks=all form=std>
  <p>All opening quotation marks converted to &odq; all closing
    quotation marks converted to &cdq;.
</editorialDecl>
```

Part: auxiliary tag set for TEI headers
Member of classes: declarable
DTD file: teihdr2
Data description:
Occurs within: encodingDesc
May contain: correction hyphenation interpretation normalization p quotation segmentation stdVals
Declaration:

```
<!ELEMENT editorialDecl - O  ( p+ | ((correction |
                             normalization | quotation |
                             hyphenation | interpretation |
                             segmentation | stdVals)+, p*))    >

<!ATTLIST editorialDecl      %a.global
                             %a.declarable                     >
```

Discussed in: 5.3.3 ('The Editorial Practices Declaration') on p. 96; 5.3 ('The Encoding Description') on p. 93; 23.3.2 ('Declarable Elements') on p. 552

education

<education> (i.e. Education) contains a brief prose description of the educational background of a participant.

Attributes: [None: global and inherited attributes only.]
Example:

```
<education>Left school at age 16
```

Part: auxiliary tag set for corpora and collections
Member of classes: demographic
DTD file: teicorp2

Data description: May contain character data and phrase-level elements.
Occurs within: person personGrp
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT education     - O  (%phrase.seq)                    >
<!ATTLIST education          %a.global                        >
```

Discussed in: 23.2.2 ('The Participants Description') on p. 545

`<eg>` contains a single example demonstrating the use of an element or attribute. If the example contains SGML markup, it must be enclosed within a marked CDATA section. `eg`

Attributes: [None: global and inherited attributes only.]
This example shows how an `<eg>` element may contain a CDATA marked section.
Example:

```
<exemplum>
<p>The <gi>term</gi> element may be used to mark any
technical term:</p>
<eg><![ CDATA [
This <term>recursion</> is giving me a headache.
]]>
</eg>
</exemplum>
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: `teitsd2`
Data description:
Occurs within: attDef etym exemplum
May contain: #PCDATA
Declaration:

```
<!ELEMENT eg            - -  (#PCDATA)                        >
<!ATTLIST eg                 %a.global                        >
```

Discussed in: 27.1 ('The TagDoc Documentation Element') on p. 603; 27.1.1 ('The AttList Documentation Element') on p. 605

`<eg>` (i.e. example (exempli gratia)) (in a dictionary) contains an example text containing at least one occurrence of the word form, used in the sense being described; examples may be quoted from (named) authors or contrived. `eg`

Attributes: [None: global and inherited attributes only.]
In some dictionaries the quoted example may be followed by a bibliographic citation for the source; this citation may be tagged with the tags described in section 6.10 ('Bibliographic Citations and References') on p. 162. The quotation and the indication of its source should be enclosed in a `<cit>` element.
Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, dictionaryTopLevel
DTD file: `teidict2`
Data description: May contain a quotation, pronunciation, definition, or translation information.
Occurs within: attDef entry etym exemplum hom re sense trans
May contain: cit q quote [May include at any level: %m.dictionaryParts %m.formPointers]
Declaration:

```
<!ELEMENT eg            - O  (q | quote | cit)+  +(%m.dictionaryParts
                                                 |
                                                 %m.formPointers)
                                                                >
```

```
<!ATTLIST eg                    %a.global
                                %a.dictionaries                 >
```

Discussed in: 12.3.5.1 ('Examples') on p. 290

eLeaf

<eLeaf> (i.e. Leaf of an embedding tree.) provides explicitly for a leaf of an embedding tree, which may also be encoded with the <eTree> element.

Attributes:

label gives a label for a leaf of an embedding tree.
Data type: CDATA
Value: A character string.
Default: #IMPLIED

value provides the value of an embedding leaf, which is a feature structure or other analytic element.
Data type: IDREF
Value: A valid identifier of a feature structure or other analytic element.
Default: #IMPLIED

Example:

```
<eLeaf label='with' value=fswith>
```

The <eTree> tag may be used if the encoder does not wish to distinguish by name between nonleaf and leaf nodes in embedding trees; they are distinguished by their arrangement.

Part: additional tag set for graphs, networks, and trees
Member of classes:
DTD file: teinet2
Data description: empty
Occurs within: eTree triangle
May contain: [none] [May include at any level: %m.dictionaryParts %m.formPointers]
Declaration:

```
<!ELEMENT eLeaf         - O  EMPTY                              >

<!ATTLIST eLeaf              %a.global
          label              CDATA               #IMPLIED
          value              IDREF               #IMPLIED       >
```

Discussed in: 21.3 ('Another Tree Notation') on p. 517

elemDecl

<elemDecl> (i.e. element declaration) contains the full form of an SGML declaration for the element documented using the reference concrete syntax.

Attributes: [None: global and inherited attributes only.]

The contents of this element are usually surrounded by a CDATA marked section, which prevents the markup-declaration-open delimiter ("<!") at the beginning of the element content from being recognized by the SGML parser (and thus causing an error).

Example:

```
<elemDecl>
<![ CDATA [
<!ELEMENT list - - (head?, ((item*)
                  | (head.label?, head.item?,
                  (label, item)+)))  >
]]>
</elemDecl>
```

The markup-declaration-open delimiter can however also be given as (or be interrupted by) an entity reference:

Example:

```
<elemDecl>
&lt;!ELEMENT list - - (head?, ((item*)
                  | (head.label?, head.item?,
                  (label, item)+)))  >
</elemDecl>
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: The declaration must be enclosed with a CDATA marked section, or else the opening delimiter must be given by an entity reference.
Occurs within: tagDoc
May contain: #PCDATA
Declaration:

```
<!ELEMENT elemDecl      - O  (#PCDATA)                    >

<!ATTLIST elemDecl           %a.global                    >
```

Discussed in: 27.1 ('The TagDoc Documentation Element') on p. 603

<emph> (i.e. emphasized) marks words or phrases which are stressed or emphasized for linguistic or rhetorical effect. — emph

Attributes: [None: global and inherited attributes only.]
Example:

```
You took the car and did <emph>what</emph>?!!
```

Example:

```
<q>What it all comes to is this,</q> he said. <q><emph>What
does Christopher Robin do in the morning nowadays?</emph></q>
```

Part: base tag set for common core features
Member of classes: hqphrase [and indirectly also:] phrase
DTD file: teicore2
Data description: free prose
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT emph          - -  (%paraContent;)              >

<!ATTLIST emph               %a.global                    >
```

Discussed in: 6.3.2.2 ('Emphatic Words and Phrases') on p. 125; 6.3.2 ('Emphasis, Foreign Words, and Unusual Language') on p. 124

<encodingDesc> (i.e. Encoding description) documents the relationship between an electronic text and the source or sources from which it was derived. — encodingDesc

Attributes: [None: global and inherited attributes only.]
Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:

Occurs within: teiHeader
May contain: classDecl editorialDecl fsdDecl metDecl p projectDesc refsDecl samplingDecl tagsDecl
Declaration:

```
<!ELEMENT encodingDesc  - -  (projectDesc*, samplingDecl*,
                              editorialDecl*, tagsDecl?,
                              refsDecl*, classDecl*, metDecl*,
                              fsdDecl*, p*)                    >

<!ATTLIST encodingDesc       %a.global                         >
```

Discussed in: 5.3 ('The Encoding Description') on p. 93; 5.1.1 ('The TEI Header and Its Components') on p. 78

entDoc

<entDoc> (i.e. entity documentation) formally documents a single SGML entity within an encoding scheme.

Attributes:

type indicates whether this is a general or a parameter entity.
Data type: (PE | GE)
Sample values include:

pe parameter entity

ge general entity

Default: #REQUIRED

Example:

```
<entDoc id=extPtr type=pe>
<entName>extPtr</entName>
<rs>extended-pointer expression</rs>
<desc>used as the declared value of an attribute, indicates that all
values of that attribute must be valid expressions in the TEI extended
pointer notation.
<string>'CDATA'</string>
<ptr target=SAxr1>
</entDoc>
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description:
Occurs within: tsd
May contain: desc entName equiv ptr remarks rs string
Declaration:

```
<!ELEMENT entDoc        - -  (entName, rs?, desc, remarks?,
                              string, ptr*, equiv*)            >

<!ATTLIST entDoc             %a.global
          type               (pe | ge)          #REQUIRED     >
```

Discussed in: 27.3 ('Entity Documentation ') on p. 607; 27 ('Tag Set Documentation') on p. 601

entitySet

<entitySet> (i.e. base entity set) identifies a public or private entity set whose mappings between entity names and characters are to be incorporated (perhaps with modifications) into this writing system declaration.

Attributes: [None: global and inherited attributes only.]

Reference in a WSD to an entity set makes the set of entity-name-to-character mappings defined in the entity set available for use in text to which the WSD applies. Unless the character-to-entity-name mappings are modified by the <exceptions> element, any name in the entity set referred to may be used with its standard meaning in any text to which the WSD applies.

Since standard public entity sets are not always completely explicit about the distinctions among characters, glyphs, graphemes, and allographs, the TEI will provide standard writing

system declarations which explicitly document the mappings provided by some commonly used public entity sets; it is recommended that similar documentation be provided for locally developed entity sets.
Part: auxiliary tag set for writing system declarations
Member of classes: baseStandard
DTD file: teiwsd2
Data description: Empty.
Occurs within: characters
May contain: [none]
Declaration:

```
<!ELEMENT entitySet     - O  EMPTY                          >

<!ATTLIST entitySet          %a.global
                             %a.baseStandard                >
```

Discussed in: 25.4.1 ('Base Components of the WSD') on p. 576

<entName> (i.e. entity name) contains the full name of an entity, excluding the percent sign in the case of a parameter entity. entName

Attributes: [None: global and inherited attributes only.]
Example:

```
<entName>component.seq
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: Must be a valid SGML name.
Occurs within: entDoc
May contain: #PCDATA
Declaration:

```
<!ELEMENT entName       - O  (#PCDATA)                      >

<!ATTLIST entName            %a.global                      >
```

Discussed in: 27.3 ('Entity Documentation ') on p. 607

<entry> contains a reasonably well-structured dictionary entry. entry

Attributes: [None: global and inherited attributes only.]

Like all elements, <entry> inherits an id attribute from the class *global*. No restrictions are placed on the method used to construct ids; one convenient method is to use the orthographic form of the headword, appending a disambiguating number where necessary. Identification codes are sometimes included on machine-readable tapes of dictionaries for in-house use.
Part: base tag set for printed dictionaries
Member of classes: comp.dictionaries, entries
DTD file: teidict2
Data description: Like all dictionary grouping tags, <entry> may contain character data mixed with any other elements defined in the dictionary tag set.
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage superentry view [May not appear at any level within: hom]
May contain: def eg etym form gramGrp hom note re sense trans usg xr [May include at any level: anchor]
Declaration:

```
<!ELEMENT entry         - O  (hom | sense |
                              %m.dictionaryTopLevel)+
                                          +(anchor)      >

<!ATTLIST entry              %a.global
                             %a.entries                  >
```

Discussed in: 12.1 ('Dictionary Body and Overall Structure') on p. 270; 12.2 ('The Structure of Dictionary Entries') on p. 274

entryFree

<entryFree> contains a dictionary entry which does not necessarily conform to the constraints imposed by the <entry> element.

Attributes: [None: global and inherited attributes only.]
Part: base tag set for printed dictionaries
Member of classes: comp.dictionaries, dictionaries, entries
DTD file: teidict2
Data description: May contain any dictionary elements in any combination.
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage view
May contain: #PCDATA [May include at any level: %m.dictionaryParts %m.inter %m.phrase]
Declaration:

```
<!ELEMENT entryFree     - O  (#PCDATA)          +(%m.dictionaryParts
                                                | %m.phrase |
                                                %m.inter)      >

<!ATTLIST entryFree          %a.global
                             %a.dictionaries
                             %a.entries                        >
```

Discussed in: 12.1 ('Dictionary Body and Overall Structure') on p. 270; 12.2 ('The Structure of Dictionary Entries') on p. 274

epigraph

<epigraph> (i.e. epigraph) contains a quotation, anonymous or attributed, appearing at the start of a section or chapter, or on a title page.

Attributes: [None: global and inherited attributes only.]
Example:

```
<epigraph LANG=lat>
 <cit><bibl>Lucret.</bibl>
        <quote>
        <l part=f>petere inde coronam,</>
        <l>Vnde prius nulli velarint tempora Musae.</l>
        </quote>
 </cit>
</epigraph>
```

Part: base tag set for common core features
Member of classes: divbot, divtop, tpParts
DTD file: teistr2
Occurs within: back body castList div div0 div1 div2 div3 div4 div5 div6 div7 epilogue front group lg performance prologue titlePage
May contain: bibl biblFull biblStruct camera caption castList cit entry entryFree event eTree graph kinesic l label lg lg1 list listBibl move note p pause q quote shift sound sp stage superentry tech termEntry tree u view vocal writing TEI...end
Declaration:

```
<!ELEMENT epigraph      - -  (%component.seq;)                 >

<!ATTLIST epigraph           %a.global                         >
```

Discussed in: 7.2.4 ('Content of Textual Divisions') on p. 194; 7.2 ('Elements Common to All Divisions') on p. 190; 7.5 ('Title Pages') on p. 203

epilogue

<epilogue> contains the epilogue to a drama, typically spoken by an actor out of character, possibly in association with a particular performance or venue.

Attributes: [None: global and inherited attributes only.]
Example:

```
<epilogue>
<head>Written by <name>Colley Cibber, Esq</name>
```

```
and spoken by <name>Mrs. Cibber</name>
<sp>
<lg>
<l>Since Fate has robb'd me of the hapless Youth,
<l>For whom my heart had hoarded up its truth;
<l>By all the Laws of Love and Honour, now,
<l>I'm free again to chuse, &mdash; and one of you
</lg>
<!-- ... -->
<lg><l>Suppose I search the sober Gallery; -- No,
<l>There's none but Prentices &mdash; & Cuckolds all a row:
<l>And these, I doubt, are those that make 'em so.</l>
<stage>Pointing to the Boxes.</stage>
<lg>
<l>'Tis very well, enjoy the jest:
<!-- ... -->
</sp>
</epilogue>
```

Part: base tag set for performance texts
Member of classes: dramafront [and indirectly also:] front
DTD file: teidram2
Data description: Contains optional headings, a sequence of one or more component-level elements, and an optional sequence of closing material.
Occurs within: back front
May contain: argument bibl biblFull biblStruct byline camera caption castList cit closer docAuthor docDate entry entryFree epigraph event eTree graph head kinesic l label lg lg1 list listBibl move note opener p pause q quote salute shift signed sound sp stage superentry tech termEntry trailer tree u view vocal writing
Declaration:

```
<!ELEMENT epilogue      - -  ((%m.divtop)*, (%component)+,
                              (%m.divbot)*)                    >

<!ATTLIST epilogue            %a.global                          >
```

Discussed in: 10.1.2 ('Prologues and Epilogues') on p. 230; 10.1 ('Front and Back Matter ') on p. 228

equipment

<equipment> (i.e. equipment) provides technical details of the equipment and media used for an audio or video recording used as the source for a spoken text.

Attributes: [None: global and inherited attributes only.]
Example:

```
<equipment>
   <p>"Hi-8" 8 mm NTSC camcorder with integral directional
   microphone and windshield and stereo digital sound
   recording channel.
</equipment>
```

Example:

```
<equipment>
  <p>8-track analogue transfer mixed down to 19 cm/sec audio
  tape for cassette mastering
</equipment>
```

Part: auxiliary tag set for TEI headers
Member of classes: declarable
DTD file: teihdr2
Data description:
Occurs within: recording
May contain: p
Declaration:

```
<!ELEMENT equipment     - O  (p+)                               >
```

```
<!ATTLIST equipment          %a.global
                             %a.declarable                    >
```

Discussed in: 5.2.9 ('Computer Files Composed of Transcribed Speech ') on p. 91; 23.3.2 ('Declarable Elements') on p. 552

equiv

<equiv> specifies an equivalent or comparable element in some other markup language.

Attributes:

scheme names the markup language or encoding scheme
Data type: CDATA
Value: any phrase identifying a markup language
Default: #REQUIRED

Example:

```
<equiv mlang=LaTeX>\para</>
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: If the other markup language uses SGML-significant characters, a CDATA marked section should be used.
Occurs within: attDef classDoc entDoc tagDoc
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph entry entryFree event expan eTree figure foreign formula gap gi gloss graph handShift hi kinesic l label lang lg lg1 link list listBibl m measure mentioned move name note num orig oRef oVar p pause phr ptr pRef pVar q quote ref reg rs s seg shift sic sound soCalled sp stage superentry table tag tech term termEntry text time timeRange timeStruct title tree u val view vocal w writing xptr xref TEI...end
Declaration:

```
<!ELEMENT equiv          - O  (%specialPara)                  >

<!ATTLIST equiv               %a.global
          scheme              CDATA              #REQUIRED   >
```

Discussed in: 27.1 ('The TagDoc Documentation Element') on p. 603; 27.1.1 ('The AttList Documentation Element') on p. 605; 27.2 ('Element Classes') on p. 606; 27.3 ('Entity Documentation ') on p. 607

eTree

<eTree> (i.e. embedding tree) provides an alternative to <tree> element for representing ordered rooted tree structures.

Attributes:

label gives a label for an embedding tree.
Data type: CDATA
Value: A character string.
Default: #IMPLIED

value provides the value of an embedding tree, which is a feature structure or other analytic element.
Data type: IDREF
Value: A valid identifier of a feature structure or other analytic element.
Default: #IMPLIED

Example:

```
<eTree n='ex1' label=PP>
   <eTree label=P><eLeaf label=with></eTree>
   <eTree label=NP>
      <eTree label=Art><eLeaf label=the></eTree>
      <eTree label=N><eLeaf label=periscope></eTree>
      </eTree>
   </eTree>
```

Part: additional tag set for graphs, networks, and trees
Member of classes: chunk [and indirectly also:] common

DTD file: teinet2
Data description: zero or more embedding trees, triangles, or embedding leafs.
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv eTree forest item metDecl note performance prologue q quote remarks set sic stage triangle view
May contain: eLeaf eTree triangle
Declaration:

```
<!ELEMENT eTree          - -  ((eTree | triangle | eLeaf )*)     >

<!ATTLIST eTree               %a.global
          label               CDATA                #IMPLIED
          value               IDREF                #IMPLIED      >
```

Discussed in: 21.3 ('Another Tree Notation') on p. 517

<etym> (i.e. etymology) encloses the etymological information in a dictionary entry. etym

Attributes: [None: global and inherited attributes only.]
Example:

```
<entry>
    <form><orth>publish</orth> ... </form>
    <etym>ME. <mentioned>publisshen</>, F. <mentioned>publier</>,
        L. <mentioned>publicare, publicatum</>.
        <xr>See <ref>public</>; cf. 2d <ref>-ish</>.</xr>
    </etym>
    <!-- ... -->
</entry>
(From:  Webster's Second International)
```

There is no consensus on the internal structure of etymologies, ore even on whether there is a standard internal structure. The <etym> tag accordingly simply contains prose, within which names of languages, cited words, glosses, and examples will typically be prominent. The tagging of such internal objects is optional.
Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, dictionaryTopLevel
DTD file: teidict2
Data description: May contain character data mixed with any other elements defined in the dictionary tag set.
Occurs within: entry hom re sense trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption case castList cit cl corr date dateRange dateStruct def del distinct eg emph expan figure foreign formula gap gen gi gloss gram handShift hi itype label lang lbl link list listBibl m measure mentioned mood move name note num number orig oRef oVar per phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title tns tr trans usg val view w xptr xr xref
Declaration:

```
<!ELEMENT etym           - O  (%paraContent | usg | lbl | def |
                              trans | tr | (%m.morphInfo) | eg |
                              xr)*                                >

<!ATTLIST etym                %a.global
                              %a.dictionaries                     >
```

Discussed in: 12.3.4 ('Etymological Information') on p. 289

<event> (i.e. Event) any phenomenon or occurrence, not necessarily vocalized or communicative, for example incidental noises or other events affecting communication. event

Attributes:

who supplies an identifier for the agent of the event described, if any. Its value is the identifier of a <participant> or <participant.grp> element in the TEI header.
Data type: IDREF
Value: Must identify a participant or participant group within the TEI Header
Default: %INHERITED

iterated (i.e. iterated) indicates whether or not the phenomenon is repeated.
Data type: (Y | N | U)
Sample values include:
y the phenomenon is repeated.
n the phenomenon is atomic.
u unknown or unmarked.
Default: N

desc (i.e. description) supplies a conventional representation for the phenomenon.
Data type: CDATA
Value: a description or representation of the phenomenon chosen from a semi-closed list
Default: #IMPLIED

Example:

```
<event desc="ceiling collapses">
```

Part: base tag set for spoken materials
Member of classes: comp.spoken, timed
DTD file: teispok2
Data description: empty
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage u view
May contain: [none]
Declaration:

```
<!ELEMENT event          - O  EMPTY                              >

<!ATTLIST event               %a.global
                              %a.timed
          who                 IDREF               %INHERITED

          iterated            (y | n | u)         n
          desc                CDATA               #IMPLIED        >
```

Discussed in: 11.2.7 ('Formal Definition') on p. 259; 11.2 ('Elements Unique to Spoken Texts') on p. 253; 11.2.3 ('Vocal, Kinesic, Event') on p. 256

exceptions

<exceptions> documents ways in which a writing system declaration differs from the coded character sets, base writing system declarations, and entity sets which form its bases.

Attributes: [None: global and inherited attributes only.]
Example:

```
<writingSystemDeclaration
    name='-//TEI P2: 1993//WSD TLG Beta code//en'
    date='1993-06-01'>
<!-- ... -->
<characters>
    <baseWsd name='-//TEI P2: 1993//WSD ISO 646 IRV//en'
             authority='TEI'>
    <exceptions>
       <character> ... </character>
       <character> ... </character>
       <character> ... </character>
       <!-- ... -->
    </exceptions>
</characters>
```

Part: auxiliary tag set for writing system declarations
Member of classes:
DTD file: teiwsd2
Data description: Contains a series of <character> elements, each documenting one character unit of the writing system.
Occurs within: characters

May contain: character
Declaration:

```
<!ELEMENT exceptions    - O  (character*)                    >
<!ATTLIST exceptions         %a.global                       >
```

Discussed in: 25.4.2 ('Exceptions in the WSD') on p. 577

`<exemplum>` contains a single example demonstrating the use of an element, together with optional paragraphs of commentary. exemplum

Attributes: [None: global and inherited attributes only.]
Example:

```
<exemplum>
 <p>The <gi>name</gi> element can be used for both
 personal names and place names:</p>
 <eg>
   <![ CDATA [
   <q>My dear <name type=person>Mr. Bennet</name>,</q>
   said his lady to him one day, <q>have you heard that
   <name type=place>Netherfield Park</name> is let
   at last?</q>
   ]]>
 </eg>
 <p>As shown, the <att>type</att> attribute may be used
 to distinguish the one from the other.
</exemplum>
```

Note that an explicit end-tag must be supplied for the paragraph immediately preceding the `<eg>` element within an `<exemplum>`, to prevent the `<eg>` from being mistaken for part of the paragraph.
Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description:
Occurs within: tagDoc
May contain: eg p
Declaration:

```
<!ELEMENT exemplum      - -  (p*, eg, p*)                    >
<!ATTLIST exemplum           %a.global                       >
```

Discussed in: 27.1 ('The TagDoc Documentation Element') on p. 603

`<expan>` (i.e. expansion) contains the expansion of an abbreviation. expan

Attributes:

abbr (i.e. abbreviation) gives the abbreviation in its unexpanded form.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED
Example:

```
The address is Southmoor <expan abbr='Rd'>Road</>.
```

resp (i.e. responsibility) signifies the editor or transcriber responsible for supplying the expansion of the abbreviation held as the content of the `<expan>` element.
Data type: IDREF
Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).
Default: %INHERITED
Example:

```
The address is Southmoor <expan abbrev='Rd' resp=LB>Road</>.
```

cert (i.e. certainty) signifies the degree of certainty ascribed to the expansion of the abbreviation.
Data type: CDATA
Default: #IMPLIED

type allows the encoder to classify the abbreviation according to some convenient typology.
Data type: CDATA
Value: any useful classification name, e.g. "suspension", "contraction", "brevigraph", "title", "organization", "geographic", etc.
Default: #IMPLIED
The type attribute is provided for the sake of those who wish to classify abbreviations at their point of occurrence; this may be useful in some circumstances, though usually the same abbreviation will have the same type in all occurrences. As the sample values make clear, abbreviations may be classified by the method used to construct them, the method of writing them, or the referent of the term abbreviated; the typology used is up to the encoder and should be carefully planned to meet the needs of the expected use.

This tag is the mirror image of the <abbr> tag; both allow the encoder to transcribe both an abbreviation and its expansion. In <abbr>, however, the original is transcribed as the content of the element and the expansion as an attribute value; <expan> reverses this. The choice between the two is up to the user.

The <expan> tag is not required; if appropriate, the encoder may expand abbreviations in the source text silently, without tagging them. If this is done, the TEI header should so indicate.

Part: additional tag set for common core features
Member of classes: data [and indirectly also:] phrase
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT expan          - -  (%phrase.seq;)                    >

<!ATTLIST expan                %a.global
          abbr                 CDATA                 #IMPLIED
          resp                 IDREF                 %INHERITED

          cert                 CDATA                 #IMPLIED
          type                 CDATA                 #IMPLIED      >
```

Discussed in: 6.4.5 ('Abbreviations and Their Expansions') on p. 139

extent

<extent> describes the approximate size of the electronic text as stored on some carrier medium, specified in any convenient units.

Attributes: [None: global and inherited attributes only.]

Example:

```
<extent>3200 sentences
<extent>between 10 and 20 Mb
<extent>ten 3.5 inch high density diskettes
```

Part: auxiliary tag set for TEI headers
Member of classes: biblPart
DTD file: teihdr2
Data description:
Occurs within: bibl biblFull fileDesc monogr
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT extent        - O  (%phrase.seq; )                    >
<!ATTLIST extent             %a.global                          >
```

Discussed in: 5.2.3 ('Type and Extent of File') on p. 85; 5.2 ('The File Description') on p. 80; 6.10.2.3 ('Imprint, Pagination, and Other Details') on p. 171

extFigure

<extFigure> (i.e. external figure) (in a writing system declaration) refers to a figure or illustration depicting the character form, which is stored in some declared notation external to the text.

Attributes:

notation identifies the notation in which the figure is stored.
Data type: NAME
Value: a valid SGML name associated with a given notation by means of an SGML NOTATION declaration in the document type definition.
Default: #REQUIRED

entity gives the SGML name of the external entity which contains the figure.
Data type: ENTITY
Value: a valid SGML name associated with the external entity by means of an SGML ENTITY declaration in the document type definition.
Default: #REQUIRED

An image of the character form may be stored and transmitted in any desired graphics format. The declaration of the notation may specify a local system identifier for the processor (here it is imagined that we use a program called "pddraw.exe" to process files in TIFF format), thus:

Example:

```
<!-- in the DTD: -->
<!NOTATION TIFF system 'pddraw.exe'>
<!ENTITY lcthorn system 'lcthorn.TIF' NDATA TIFF>

<!-- in the WSD itself: -->
<extFigure notation=TIFF entity='lcthorn'>
```

The graphics format might also be declared with a public identifier in addition to the system identifier; this is likely to be more comprehensible and thus more useful to users of a different system:

Example:

```
<!-- in the DTD: -->
<!NOTATION TIFF
           PUBLIC '-//XXX//NOTATION Tagged Image File Format//EN'
           'pddraw.exe'>
<!ENTITY lcthorn system 'lcthorn.TIF' NDATA TIFF>

<!-- in the WSD itself: -->
```

```
<extFigure notation=TIFF entity='lcthorn'>
```

Character shapes may be conveniently conveyed in forms other than graphics images; one might, for example, define a character shape using a font-design program such as Donald Knuth's MetaFont program (which can be used to generate fonts for processing with TEX):

Example:

```
<!-- in the DTD: -->
<!NOTATION metafont
           PUBLIC '-//DEK//NOTATION MetaFont//EN'
           'mf.exe'>
<!ENTITY lcthorn system 'lcthorn.TIF' NDATA metafont>

<!-- in the WSD itself: -->
<extFigure notation=metafont entity='lcthorn'>
```

Part: auxiliary tag set for writing system declarations
Member of classes:
DTD file: teiwsd2
Data description: Empty.
Occurs within: form formula
May contain: [none]
Declaration:

```
<!ELEMENT extFigure     - O  EMPTY                          >

<!ATTLIST extFigure          %a.global
          notation           NAME                #REQUIRED
          entity             ENTITY              #REQUIRED   >
```

Discussed in: 25.4.2 ('Exceptions in the WSD') on p. 577

f

<f> (i.e. Feature) associates a name with a value of any of several different types.

Attributes:

name provides a name for a feature.
Data type: NMTOKEN
Value: A name token.
Default: #REQUIRED

org indicates organization of given value or values as singleton, set, bag or list.
Data type: (SINGLE | SET | BAG | LIST)
Sample values include:

single indicates that the given value is a singleton.
set indicates that the given values are organized as a set.
bag indicates that the given values are organized as a bag (multiset).
list indicates that the given values are organized as a list.

Default: #IMPLIED
If the org=single attribute value is specified and more than one value for the feature is given, then only the first value is used.

rel indicates the relation between the values that are given as the content of the feature or pointed at by the fVal attribute and the actual values of the feature.
Data type: (EQ | NE | SB | NS)
Sample values include:

eq indicates that the given values are the actual values.
ne indicates that the given values are not the actual values.
sb indicates that the given values are a subset, subbag or sublist of the actual values.
ns indicates that the given values are not a subset, subbag or sublist of the actual values.

Default: EQ
If org=single, then rel=sb is equivalent to rel=eq, and rel=ns is equivalent to rel=ne.

fVal points to the id attributes of feature values.
Data type: IDREFS
Value: one or more legal SGML identifiers.
Default: #IMPLIED
May be used instead of content.

Example:

```
<f name=gender><sym value=feminine></f>
```

If content is empty and no fVal attribute is present, then value is that specified by <default>.
Content is defined as (null | (%mVal;)*) in FS3.dtd. The entity %mVal; is defined as "%aVal; | %dVal;" in FS3.dtd. %aVal; is defined as "plus | minus | any | none | dft | uncertain". %dVal; is defined as "%cVal; | fs". Finally, %cVal; is defined as "sym | nbr | msr | rate | str | vAlt | alt",
Part: additional tag set for feature structures
Member of classes:
DTD file: teifs2
Data description: Exactly one null value, or zero or more values other than null.
Occurs within: bicond cond fs fAlt fLib if
May contain: alt any dft fs minus msr nbr none null plus rate str sym uncertain vAlt
Declaration:

```
<!ELEMENT f             - O  (null | (plus | minus | any | none
                              | dft | uncertain | sym | nbr |
                              msr | rate | str | vAlt | alt |
                              fs)*)                              >

<!ATTLIST f                  %a.global
          name               NMTOKEN             #REQUIRED
          org                (single | set | bag | list)
                                                 #IMPLIED
          rel                (eq | ne | sb | ns) eq
          fVal               IDREFS              #IMPLIED       >
```

Discussed in: 16.2 ('Elementary Feature Structures: Features with Binary Values') on p. 398

factuality

<factuality> describes the extent to which the text may be regarded as imaginative or non-imaginative, that is, as describing a fictional or a non-fictional world.

Attributes:

type categorizes the factuality of the text.
Data type: (FICTION | FACT | MIXED | INAPPLICABLE)
Sample values include:

fiction the text is to be regarded as entirely imaginative
fact the text is to be regarded as entirely informative or factual
mixed the text contains a mixture of fact and fiction
inapplicable the fiction/fact distinction is not regarded as helpful or appropriate to this text

Default: #IMPLIED

Example:

```
<factuality type=fiction>
```

Example:

```
<factuality type=mixed>contains a mixture of gossip and
  speculation about real people and events
```

For many literary texts, a simple binary opposition between "fiction" and "fact" is naïve in the extreme; this parameter is not intended for purposes of subtle literary analysis, but as a simple means of characterising the claimed fictiveness of a given text. No claim is made that works characterised as "fact" are in any sense "true".
Part: auxiliary tag set for corpora and collections

Member of classes:
DTD file: teicorp2
Data description: Usually empty, unless some further clarification of the type attribute is needed, in which case it may contain running prose
Occurs within: textDesc
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT factuality    - O  (%phrase.seq)                    >

<!ATTLIST factuality         %a.global
          type               (fiction | fact | mixed |
                             inapplicable)       #IMPLIED      >
```

Discussed in: 23.2.1 ('The Text Description') on p. 541

fAlt

<fAlt> (i.e. Feature alternation) provides alternative features for a feature structure or other feature alternation.

Attributes:

mutExcl indicates whether values are mutually exclusive.
Data type: (Y | N)
Sample values include:
Y indicates that the values are mutually exclusive.
N indicates that the values are not mutually exclusive.
Default: #IMPLIED

Example:

```
<fAlt mutExcl=Y>
  <f name=gender><sym value=masculine>
  <f name=gender><sym value=neuter>
  </fAlt>
```

Part: additional tag set for feature structures
Member of classes:
DTD file: teifs2
Data description: Two or more features, feature structures or feature alternations.
Occurs within: bicond cond fs fAlt fLib if
May contain: f fs fAlt
Declaration:

```
<!ELEMENT fAlt          - -  ((f | fs | fAlt), (f | fs |
                             fAlt)+)                       >

<!ATTLIST fAlt               %a.global
          mutExcl            (Y | N)          #IMPLIED     >
```

Discussed in: 16.7 ('Alternative Features and Feature Values') on p. 413

fDecl

<fDecl> (i.e. feature declaration) declares a single feature, specifying its name, organization, range of allowed values, and optionally its default value.

Attributes:

name indicates the name of the feature being declared; matches the name attribute of <f> elements in the text.
Data type: NMTOKEN
Value: any string of characters
Default: #REQUIRED
org (i.e. organization) specifies the organizing discipline of the feature value.
Data type: (UNIT | SET | BAG | LIST)
Sample values include:
unit unitary atomic value

set set value (unordered, no duplicates)
bag bag value (unordered, may have duplicates)
list list value (ordered, may have duplicates)

Default: UNIT

Part: auxiliary tag set for feature system declarations
Member of classes:
DTD file: teifsd2
Data description: May contain character data and phrase-level elements.
Occurs within: fsDecl
May contain: fDescr vDefault vRange
Declaration:

```
<!ELEMENT fDecl         - -  (fDescr?, vRange, vDefault?)       >

<!ATTLIST fDecl              %a.global
          name               NMTOKEN             #REQUIRED
          org                (unit | set | bag | list)
                                                 unit           >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

fDescr

<fDescr> (i.e. feature description (in FSD)) describes in prose what is represented by the feature being declared and its values.

Attributes: [None: global and inherited attributes only.]
Part: auxiliary tag set for feature system declarations
Member of classes:
DTD file: teifsd2
Data description: May contain character data, phrase-level elements, and inter-level elements.
Occurs within: fDecl
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT fDescr        - O  (%paraContent;)                    >

<!ATTLIST fDescr             %a.global                          >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

figDesc

<figDesc> (i.e. Description of Figure) contains a brief prose description of the appearance or content of a graphic figure, for use when documenting an image without displaying it.

Attributes: [None: global and inherited attributes only.]
This element is intended for use as an alternative to the content of its parent <figure> element; for example, to display when the image is required but the equipment in use cannot display graphic images. It may also be used for indexing or documentary purposes.
Part: additional tag set for
Member of classes:
DTD file: teifig2
Data description: May contain character data and phrase-level elements.
Occurs within: figure
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT figDesc       - -  (%paraContent;)                    >

<!ATTLIST figDesc            %a.global                          >
```

Discussed in: 22.3 ('Specific Elements for Graphic Images') on p. 530

figure

<figure> (in a writing system declaration) contains an image of a character form, stored in-line in some declared notation.

Attributes:

notation identifies the notation in which the figure is encoded.

Data type: NAME

Value: a valid SGML name associated with a given notation by means of an SGML NOTATION declaration in the document type definition.

Default: #REQUIRED

Example:

```
<!-- in the DTD -->
<!NOTATION charcell
           PUBLIC '-//Anonymous//NOTATION
                   8x14 character cell format//en' >

<!-- 'CHARCELL' is a simple-minded notation for transcribing  -->
<!-- character images in the form used by the old IBM PC:     -->
<!-- an eight-by-fourteen pixel character cell.  Each pixel   -->
<!-- is represented by a period (if off) or an X (if on).     -->
<!-- Newlines are ignored, but for obvious reasons it's       -->
<!-- convenient to have one after each line of pixels.        -->

<!-- N.B.  This 'notation' is given as an example; it is not  -->
<!-- (repeat, NOT) recommended for serious use.               -->

<!-- in the WSD itself: -->
<character class='lexical'>
<desc>Latin lowercase letter thorn</desc>
<form entityStd='thorn' entityLoc='t'>
<figure notation=charcell>
 ........
 ........
 XXX.....
 .XX.....
 .XX.....
 .XXXXX..
 .XX..XX.
 .XX..XX.
 .XX..XX.
 .XXXXX..
 .XX.....
 .XX.....
 XXXX....
 ........
</figure>
</form>
</character>
```

Inline storage of non-SGML data may be convenient for a variety of reasons, but complications occasionally ensue if any portion of the data can be interpreted by the SGML parser as ending some open SGML element. For this reason, inline storage should be used only for extremely simple notations such as the one shown, or by users well versed in dealing with SGML and graphics. In general, external storage of figures is recommended because less error prone.

Part: auxiliary tag set for writing system declarations

Member of classes:

DTD file: teiwsd2

Data description: May contain non-SGML character data.

Occurs within: form formula

May contain: [none]
Declaration:

```
<!ELEMENT figure        - -  CDATA                          >
<!ATTLIST figure             %a.global
          notation           NAME                #REQUIRED    >
```

Discussed in: 25.4.2 ('Exceptions in the WSD') on p. 577

<figure> indicates the location of a graphic, illustration, or figure. figure

Attributes:

entity names the external entity within which the graphic image of the figure is stored.
Data type: ENTITY
Value: the entity name of some external entity declared either as a SUBDOC entity or as an entity using a non-SGML notation.
Default: #IMPLIED
This attribute may be omitted if for some reason no electronic form of the image is provided.

Example:

```
<figure entity='Fig1'>
<head>Figure One: The View from the Bridge</head>
<figDesc>A Whistleresque view showing four
or five sailing boats in the foreground, and a
series of buoys strung out between them.</figDesc>
</figure>
```

The end-tag must be supplied, even if the element has no content.
Part: additional tag set for formulae
Member of classes: inter
DTD file: teifig2
Data description:
Occurs within: add admin camera caption case cell colloc corr country damage def desc descrip docEdition emph equiv etym figDesc foreign form formula fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting mood note number orth otherForm p per pos pron q quote rdg ref region rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref
May contain: figDesc head p text
Declaration:

```
<!ELEMENT figure        - -  (head?, p*, figDesc?, text?)   >
<!ATTLIST figure             %a.global
          entity             ENTITY              #IMPLIED    >
```

Discussed in: 22 ('Tables, Formulae, and Graphics') on p. 523

<fileDesc> (i.e. File Description) contains a full bibliographic description of an electronic file. fileDesc

Attributes: [None: global and inherited attributes only.]

The major source of information for those seeking to create a catalogue entry or bibliographic citation for an electronic file. As such, it provides a title and statements of responsibility together with details of the publication or distribution of the file, of any series to which it belongs, and detailed bibliographic notes for matters not addressed elswhere in the header. It also contains a full bibliographic description for the source or sources from which the electronic text was derived.

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:
Occurs within: teiHeader
May contain: editionStmt extent notesStmt publicationStmt seriesStmt sourceDesc titleStmt
Declaration:

```
<!ELEMENT fileDesc      - -  (titleStmt, editionStmt?, extent?,
                              publicationStmt, seriesStmt?,
                              notesStmt?, sourceDesc+ )          >

<!ATTLIST fileDesc           %a.global                           >
```

Discussed in: 5.2 ('The File Description') on p. 80; 5.1.1 ('The TEI Header and Its Components') on p. 78

files

<files> specifies the name of the operating system file(s) within which this markup component is declared.

Attributes:

names supplies the names of one or more files.
Data type: CDATA
Value: a file identifier
Default: #IMPLIED

Example:

```
<files names=TEIdict2>
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: Empty element.
Occurs within: classDoc tagDoc
May contain: [none]
Declaration:

```
<!ELEMENT files         - O  EMPTY                              >

<!ATTLIST files              %a.global
          names              CDATA                #IMPLIED     >
```

Discussed in: 27.1 ('The TagDoc Documentation Element') on p. 603

firstLang

<firstLang> (i.e. First language) specifies the first language of a participant.

Attributes: [None: global and inherited attributes only.]
Example:

```
<firstLang>French
```

Part: auxiliary tag set for corpora and collections
Member of classes: demographic
DTD file: teicorp2
Data description:
Occurs within: person personGrp
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT firstLang     - O  (%phrase.seq)                      >

<!ATTLIST firstLang          %a.global                          >
```

Discussed in: 23.2.2 ('The Participants Description') on p. 545

fLib

<fLib> (i.e. Feature library) assembles library of feature elements.

Attributes:

type indicates type of feature library (i.e., what kind of features it contains).
Data type: CDATA
Value: Character string, e.g. *word features*.
Default: #IMPLIED

Example:

```
<fLib type='agreement features'>
  <f name=person id=P1 fVal='sfirst'></f>
  <f name=person id=P2 fVal='ssecond'></f>
<!-- ... -->
  <f name=number id=Ns fVal='ssing'></f>
  <f name=number id=Np fVal='splur'></f>
<!-- ... -->
  </fLib>
```

The entity %fFamily; is used for "f | fAlt" in FS3.dtd.

Part: additional tag set for feature structures
Member of classes:
DTD file: teifs2
Data description: Zero or more features or alternations of features.
Occurs within: [none]
May contain: f fAlt
Declaration:

```
<!ELEMENT fLib          - -  ((f | fAlt)*)                    >
<!ATTLIST fLib               %a.global
          type               CDATA                #IMPLIED      >
```

Discussed in: 16.3 ('Feature, Feature-Structure and Feature-Value Libraries') on p. 400

foreign

<foreign> (i.e. foreign) identifies a word or phrase as belonging to some language other than that of the surrounding text.

Attributes:

lang (i.e. language) identifies the language of the word or phrase marked.
Data type: IDREF
Value: contains a language code (associated with some writing system declaration) for the language in question; where applicable, the codes of ISO 639 should be used.
Default: #IMPLIED
It is strongly recommended that the lang attribute be consistently specified on all <foreign> elements.

Example:

```
This is heathen Greek to you still?
Your <foreign lang=lat>lapis philosophicus</foreign>?
```

This element is intended for use only where no other element is available to mark the phrase or words concerned. The global lang attribute should be used in preference to this element where it is intended to mark the language of the whole of some text element.

The <distinct> element may be used to identify phrases belonging to sublanguages or registers not generally regarded as true languages.

Part: base tag set for common core features
Member of classes: hqphrase [and indirectly also:] phrase
DTD file: teicore2
Data description: free prose
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref

Declaration:

```
<!ELEMENT foreign       - -  (%paraContent;)                    >

<!ATTLIST foreign            %a.analysis
                             %a.linking
                             %a.terminology
          id                 ID                  #IMPLIED
          n                  CDATA               #IMPLIED
          rend               CDATA               #IMPLIED
          lang               IDREF               #IMPLIED       >
```

Discussed in: 6.3.2.1 ('Foreign Words or Expressions') on p. 124

forename

<forename> (i.e. forename) contains a forename, given or baptismal name.

Attributes: [None: global and inherited attributes only.]

Example:

```
<persName>
  <roleName>Ex-President</roleName>
  <foreName>George</foreName>
  <surname>Bush</surname>
</persName>
```

Part: additional tag set for names and dates

Member of classes: personPart [and indirectly also:] names

DTD file: teind2

Occurs within: persName

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT forename      - -  (%phrase.seq;)                     >

<!ATTLIST forename           %a.global
                             %a.personPart                     >
```

Discussed in: 20.1 ('Personal Names') on p. 488

forest

<forest> provides for groups of rooted trees.

Attributes:

type identifies the type of the forest.
Data type: CDATA
Value: A character string.
Default: #IMPLIED

Part: additional tag set for graphs, networks, and trees

Member of classes:

DTD file: teinet2

Data description: One or more trees, embedding trees, or underspecified embedding trees (triangles).

Occurs within: forestGrp

May contain: eTree tree triangle

Declaration:

```
<!ELEMENT forest        - -  ((tree | eTree | triangle)+)       >

<!ATTLIST forest             %a.global
          type               CDATA               #IMPLIED       >
```

Discussed in: 21.3 (‘Another Tree Notation’) on p. 517

<forestGrp> (i.e. Forest group) provides for groups of forests. forestGrp

Attributes:

type identifies the type of the forest group.
Data type: CDATA
Value: A character string.
Default: #IMPLIED

Example:

```
<!--None provided. -->
```

Part: additional tag set for graphs, networks, and trees
Member of classes:
DTD file: teinet2
Data description: One or more forests.
Occurs within: [none]
May contain: forest
Declaration:

```
<!ELEMENT forestGrp     - -  ((forest)+)                      >

<!ATTLIST forestGrp           %a.global
          type                CDATA               #IMPLIED      >
```

Discussed in: 21.3 (‘Another Tree Notation’) on p. 517

<form> (i.e. form information group) groups all the information on the written and spoken forms of one headword. form

Attributes:

type classifies form as simple, compound, etc.
Data type: CDATA
Sample values include:

simple single free lexical item
lemma the headword itself
variant a variant form
compound word formed from simple lexical items
derivative word derived from headword
inflected word in other than usual dictionary form
phrase multiple-word lexical item

Default: #IMPLIED

Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, dictionaryTopLevel, formInfo
DTD file: teidict2
Data description: May contain any of a variety of element.
Occurs within: character entry form hom re sense superentry trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign form formula gap gi gloss handShift hi hyph label lang lbl link list listBibl m measure mentioned move name note num orig orth oRef oVar phr pron ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage syll table tag tech term text time timeRange timeStruct title usg val view w xptr xref
Declaration:

```
<!ELEMENT form          - -  (%m.formInfo | %paraContent)+     >

<!ATTLIST form                %a.global
                              %a.dictionaries
          type                CDATA               #IMPLIED      >
```

Discussed in: 12.3.1 (‘Information on Written and Spoken Forms’) on p. 279

<form> (i.e. letter form) identifies one letter form taken by a particular character in a writing system declaration. form

Attributes:

string gives the byte string used to encode the letter form in the text.

Data type: CDATA

Value: any string of characters (often a single byte)

Default: #IMPLIED

Example:

```
<form string='a/'>
   <desc>lowercase Greek alpha with acute accent</desc>
</form>
```

If the character is encoded only using entity references, then the value of string should be '' (the empty string).

In coded character sets which use character-set shifting (e.g. JIS 0208), the string attribute should typically contain the required shift characters, in order to render the value unambiguous. In such a case, there is no expectation that every occurrence of the character will be immediately preceded by the shift sequence; processing software is responsible for understanding the shift mechanism and acting accordingly.

The same string value may not appear on more than one <form> elements (except the empty string), unless each occurrence is associated with a different coded character set.

codedCharSet (i.e. coded character set) specifies which base coded character set the string value occurs in.

Data type: IDREF

Value: a reference to the SGML identifier of a <codedCharSet> element in the current writing system declaration.

Default: #IMPLIED

If more than one <codedCharSet> is specified as a base component of the writing system declaration, then it is expected that character-set shifting is in use, as described in ISO 2022 or some equivalent. In this case, each <form> element which has a value for the string attribute should also identify, by means of the codedCharSet attribute, which identifies which coded character set actually contains the string in question. Proper shifting among character sets is the responsibility of the user.

entityStd (i.e. standard entity name) gives the name of one or more entities defined for this character form in some standard entity set(s).

Data type: ENTITIES

Value: One or more valid SGML entity names declared in the document type definition of the WSD; the entity must also be included in an entity set mentioned in an <entitySet> declaration in the current writing system declaration or in some base writing system referred to by a <baseWsd> element.

Default: #IMPLIED

Example:

```
<form entityStd='thorn'>
  <desc>lowercase Old English/Icelandic thorn</desc>
<form>
```

If the same letter form is defined by more than one public entity set, more than one value may appear in this attribute.

The same entity name may not appear in the entityStd or entityLoc attributes of more than one <form> element.

entityLoc (i.e. local entity name) gives one or more entity names used locally for this character form.

Data type: ENTITIES

Value: One or more valid SGML entity names declared in the document type definition of the WSD; the entity must also be included in an entity set mentioned in an <entitySet> declaration in the current writing system declaration or in some base writing system referred to by a <baseWsd> element.

Default: #IMPLIED
Example:

```
<form entityStd='thorn' entityLoc='t'>
  <desc>lowercase Old English/Icelandic thorn</desc>
  <note>The standard entity name is 'thorn'; the local entity 't'
        is used for brevity and legibility.</note>
<form>
```

The same entity name may not appear in the entityStd or entityLoc attributes of more than one <form> element.

ucs-4 (i.e. universal-character-set code) gives the position of the character form in the thirty-two bit "universal character set" defined by ISO 10646.
Data type: CDATA
Value: one or more sets of two or four two-digit hexadecimal numbers giving a valid ISO 10646 code point for the character form; for legibility the two-digit hexadecimal numbers should be separated by hyphens. If more than one UCS-4 code is associated with a given character form, the two UCS-4 codes should be given separated by blanks. If the character form is associated with a sequence of UCS-4 codes (e.g. a base character followed by one or more non-spacing diacritics), then the components of the sequence should be separated by '+'.
Default: #IMPLIED
The same UCS-4 code (or sequence) may not appear within more than one <character> element within the writing system declaration. It may however appear on several forms of the same character.
Multiple UCS-4 codes can be given for a single character; this allows sequences treated as distinct by ISO 10646 to be documented as referring to a single "character" as defined by the WSD (e.g. "lowercase a-umlaut" and "lowercase a" plus "umlaut").
If a single UCS-4 code is to be treated as relating to two distinct "characters" as defined by the WSD (e.g. to reverse the effects of Han unification on some character), then one of the <character> elements should be associated with the UCS-4 code in the normal way, and the others should call attention to the relevant UCS-4 code by a comment in a <note> element.

afiicode (i.e. AFII code) gives one or more codes associated with this letter form by the Association for Font Information Interchange.
Data type: CDATA
Value: any valid AFII identifier.
Default: #IMPLIED
The AFII tables are designed as an inventory of *glyphs* (identifiably distinct shapes, leaving differences of font design out of account—one character may be associated with several glyphs, and each glyph with items in several different fonts). Because the same glyph may be associated with more than one character (in some fonts, for example, the lowercase letter L and the digit 1 share the same glyph), the value of afiiCode is used for informational purposes only and need not be unique within the writing system declaration.

The <form> element documents one form of a character; in most cases, there will be only one. If more than one form is given, in general, they are to be regarded as free variants of the character unless otherwise specified in the notes.

The distinction between <character> and <form> makes it possible to distinguish, in an encoding, among different letter forms (which may have historical, aesthetic, linguistic, or other significance) without having to claim that the different forms constitute different "characters" in any normal sense. (Using the technical terms occasionally encountered, the <form> element can be used to record each *allograph* of a given character or *grapheme*.) The concepts of "character" and "letter form", however, vary from analyst to analyst; the decision to treat a given set of forms as a single character or as a set of characters is not always obvious, and may require the application of considerable learning and judgement. The <note> element should be used to record the reasoning behind any particularly difficult decision.

Part: auxiliary tag set for writing system declarations
Member of classes:
DTD file: teiwsd2
Data description: May contain a series of description element, optionally one or more figure elements showing the character form in question, and optionally a series of notes.
Occurs within: character superentry
May contain: desc extFigure figure note
Declaration:

```
<!ELEMENT form          - O  (desc+, (figure | extFigure)*,
                              note*)                          >

<!ATTLIST form                %a.global
          string              CDATA                #IMPLIED
          codedCharSet        IDREF                #IMPLIED
          entityStd           ENTITIES             #IMPLIED
          entityLoc           ENTITIES             #IMPLIED
          ucs-4               CDATA                #IMPLIED
          afiicode            CDATA                #IMPLIED    >
```

Discussed in: 25.4.2 ('Exceptions in the WSD') on p. 577

formula

<formula> contains a mathematical or other formula.

Attributes:

notation supplies the name of a previously defined notation used for the content of the element.
Data type: %FORMULANOTATIONS
Value: The name of a notation for which a formal SGML notation declaration has been provided in the document type declaration.
Default: #REQUIRED

Part: base tag set for formulae
Member of classes: phrase
DTD file: teifig2
Data description: The content model for this element is specified by the parameter entity *formulaContent*, the default value of which is "CDATA".
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: desc extFigure figure note
Declaration:

```
<!ELEMENT formula       - -                                   >

<!ATTLIST formula             %a.global
          notation            %formulaNotations   #REQUIRED   >
```

Discussed in: 22.2 ('Formulae') on p. 527

front

<front> (i.e. front matter) contains any prefatory matter (headers, title page, prefaces, dedications, etc.) found before the start of a text proper.

Attributes: [None: global and inherited attributes only.]
Example:

```
<front>
<epigraph lang=LA>
Nam Sibyllam quidem Cumis ego ipse oculis meis
vidi in ampulla pendere, et cum illi pueri dicerent:
<q lang=GRC>Sibylla ti weleis</q>; respondebat
illa: <q lang=GRC>apowanein welo.</q></epigraph>
<div type=dedication>
<p>For Ezra Pound <q lang=IT>il miglior fabbro.</q></p>
</front>
```

Example:

```
<front>
<div type=dedication>
<p>To our three selves
</div>
<div type=preface>
<head>Author's Note</head>
<p>All the characters in this book are purely imaginary, and if the
author has used names that may suggest a reference to living persons
she has done so inadvertently.
 ...
</div>
</front>
```

Part: base tag set for common core features
Member of classes: declaring
DTD file: teifron2
Occurs within: text
May contain: argument byline div div1 docAuthor docDate epigraph epilogue head opener performance prologue salute set signed titlePage
Declaration:

```
<!ELEMENT front         - O  ( (%m.front;)*, ( ( (%m.divtop;),
                             (%m.divtop; | titlePage)*) | (
                             (div), (div | (%m.front;) )*) | (
                             (div1), (div1 | (%m.front;) )*) )?
                             )                                  >

<!ATTLIST front              %a.global
                             %a.declaring                       >
```

Discussed in: 7.5 ('Title Pages') on p. 203; 7 ('Default Text Structure') on p. 183

<fs> (i.e. Feature structure) analyzes a collection of features and feature alternations as a structural unit. fs

Attributes:

type provides a type for a feature structure.
Data type: CDATA
Value: Character string, e.g. *word structure*.
Default: #IMPLIED

feats pointer to features.
Data type: IDREFS
Value: list of individual IDs for features.
Default: #IMPLIED
May be used instead of having features as content.

rel indicates the relation of the given content to the actual content or value of the feature structure.
Data type: (EQ | NE | SB | NS)
Sample values include:

eq indicates that the actual content is that given.
ne indicates that the actual content is not that given.

sb indicates that the actual content is subsumed by the given content.

ns indicates that the actual content is not subsumed by the given content.

Default: SB

The <fs> element is the only one for which the default rel attribute value is *sb*. For all others, it is *eq*.

Example:

```
<fs type='agreement structure' rel=ns>
   <f name=person><sym value=third></f>
   <f name=number><sym value=singular></f>
   </fs>
```

Part: additional tag set for feature structures
Member of classes: complexVal [and indirectly also:] featureVal
DTD file: teifs2
Data description: Zero or more features or (feature) alternations.
Occurs within: bicond cond f fsLib fAlt if vAlt vDefault vRange
May contain: alt f fAlt
Declaration:

```
<!ELEMENT fs            - -  ((f | fAlt | alt)*)                >

<!ATTLIST fs                 %a.global
          type               CDATA               #IMPLIED
          feats              IDREFS              #IMPLIED
          rel                (eq | ne | sb | ns) sb              >
```

Discussed in: 16.2 ('Elementary Feature Structures: Features with Binary Values') on p. 398

fsConstraints

<fsConstraints> (i.e. feature-structure constraints) specifies constraints on the content of well formed feature structures.

Attributes: [None: global and inherited attributes only.]
Part: auxiliary tag set for feature system declarations
Member of classes:
DTD file: teifsd2
Data description: May contain a series of conditional or biconditional elements.
Occurs within: fsDecl
May contain: bicond cond
Declaration:

```
<!ELEMENT fsConstraints - -  (cond | bicond)*                   >

<!ATTLIST fsConstraints      %a.global                          >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

fsdDecl

<fsdDecl> (i.e. FSD (feature-system declaration) declaration) identifies the feature system declaration which contains definitions for a particular type of feature structure.

Attributes:

type identifies the type of feature structure documented in the FSD; this will be the value of the type attribute on at least one feature structure.
Data type: CDATA
Value: any string of characters.
Default: #REQUIRED

fsd (i.e. feature-system declaration) specifies the external entity containing the feature system declaration; an entity declaration in the document's DTD subset must associate the entity name with a file on the system.
Data type: ENTITY
Value: an entity name (and therefore a valid SGML identifier)
Default: #REQUIRED

Example:

```
<encodingDesc>
      <!-- ... -->
      <fsdDecl type=GPSG     fsd=fsd.gazdar>
      <fsdDecl type=entry    fsd=fsd.lexicon>
      <fsdDecl type=subentry fsd=fsd.lexicon>
      <!-- ... -->
 </encodingDesc>
```

Part: additional tag set for feature structures
Member of classes:
DTD file: teihdr2
Data description: Empty.
Occurs within: encodingDesc
May contain: [none]
Declaration:

```
<!ELEMENT fsdDecl       - O  EMPTY                          >

<!ATTLIST fsdDecl            %a.global
          type               CDATA               #REQUIRED
          fsd                ENTITY              #REQUIRED  >
```

Discussed in: 5.3.7 ('The Feature System Declaration') on p. 105; 26 ('Feature System Declaration') on p. 589

<fsDecl> (i.e. feature structure declaration) declares one type of feature structure. — fsDecl

Attributes:

type gives a name for the type of feature structure being declared.
Data type: CDATA
Value: any convenient string of characters.
Default: #REQUIRED

baseType gives the name of the feature structure type from which this type inherits features and constraints; if this type declares a feature with the same name as a feature of the base type, the definition within this <fsDecl> overrides the inherited definition. The <fsConstraints> are inherited only if this <fsDecl> does not specify any; otherwise the constraints in this <fsDecl> override. When no baseType is specified, no features or constraints are inherited.
Data type: CDATA
Value: any convenient string for use as a name.
Default: #IMPLIED

Part: auxiliary tag set for feature system declarations
Member of classes:
DTD file: teifsd2
Data description: Contains a description (optional), a series of feature declarations, and an optional set of feature structure constraints.
Occurs within: teiFsd2
May contain: fsConstraints fsDescr fDecl
Declaration:

```
<!ELEMENT fsDecl        - -  (fsDescr?, fDecl+, fsConstraints?)
                                                            >

<!ATTLIST fsDecl             %a.global
          type               CDATA               #REQUIRED
          baseType           CDATA               #IMPLIED   >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

<fsDescr> (i.e. feature system description (in FSD)) describes in prose what is represented by the type of feature structure declared in the enclosing <fsDecl>. — fsDescr

Attributes: [None: global and inherited attributes only.]

Part: auxiliary tag set for feature system declarations
Member of classes:
DTD file: teifsd2
Data description: May contain character data, phrase-level elements, and inter-level elements.
Occurs within: fsDecl
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT fsDescr       - O  (%paraContent;)                    >
<!ATTLIST fsDescr            %a.global                          >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

fsLib

<fsLib> (i.e. Feature-structure library) assembles library of feature structure elements.

Attributes:

type indicates type of feature-structure library (i.e., what type of feature structures it contains).
Data type: CDATA
Value: Character string, e.g. *word structure library*.
Default: #IMPLIED

Example:

```
<fsLib type='agreement structure library'>
  <fs type='agreement structure' id=P1Ns feats='P1 Ns'></fs>
  <fs type='agreement structure' id=P1Np feats='P1 Np'></fs>
<!-- ... -->
  </fsLib>
```

If a <vAlt> member of an <fsLib> tag does not consist of <fs> tags, the result is a semantic error.
Part: additional tag set for feature structures
Member of classes:
DTD file: teifs2
Data description: Zero or more feature structures or alternations of feature structures (expressed as <vAlt> elements, see remarks).
Occurs within: [none]
May contain: fs vAlt
Declaration:

```
<!ELEMENT fsLib         - -  ((fs | vAlt)*)                     >
<!ATTLIST fsLib              %a.global
          type               CDATA                #IMPLIED     >
```

Discussed in: 16.3 ('Feature, Feature-Structure and Feature-Value Libraries') on p. 400

funder

<funder> (i.e. Funding body) specifies the name of an individual, institution, or organization responsible for the funding of a project or text.

Attributes: [None: global and inherited attributes only.]
Example:

```
<funder>The National Endowment for the Humanities, an independent
        federal agency</>
<funder>Directorate General XIII of the Commission of the
        European Communities</>
<funder>The Andrew W. Mellon Foundation</>
<funder>The Social Sciences and Humanities Research Council of Canada</>
```

Funders provide financial support for a project; they are distinct from *sponsors*, who provide intellectual support and authority.

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description: Any string of characters.
Occurs within: titleStmt
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT funder        - O  ( %phrase.seq; )                  >
<!ATTLIST funder             %a.global                         >
```

Discussed in: 5.2.1 ('The Title Statement') on p. 82

<fvLib> (i.e. Feature-value library) assembles library of feature value elements. fvLib

Attributes:

type indicates type of feature-value library (i.e., what type of feature values it contains).
Data type: CDATA
Value: Character string, e.g. *symbolic values*.
Default: #IMPLIED

Example:

```
<fvLib type='symbolic values'>
  <sym id=sfirst value=first>
  <sym id=ssecond value=second>
<!-- ... -->
  <sym id=ssing value=singular>
  <sym id=splur value=plural>
<!-- ... -->
  </fvLib>
```

Content defined as ((%bVal;)? & (%cVal; | ptr | join)*) in FS3.dtd.
If a <vAlt> member of an <fvLib> tag contains <fs> tags, the result is a semantic error.
Part: additional tag set for feature structures
Member of classes:
DTD file: teifs2
Data description: At most one of each of the *unique* feature values; and zero or more *nonunique* feature values, except feature structures (which should be placed in feature-structure libraries).
Occurs within: [none]
May contain: any dft minus msr nbr none null plus rate str sym uncertain vAlt
Declaration:

```
<!ELEMENT fvLib         - -  ((plus | minus | any | none | dft
                              | uncertain | null | sym | nbr |
                              msr | rate | str | vAlt)*)        >
<!ATTLIST fvLib              %a.global
          type               CDATA                #IMPLIED      >
```

Discussed in: 16.3 ('Feature, Feature-Structure and Feature-Value Libraries') on p. 400

<fw> (i.e. forme work) contains a running head (e.g. a "header", "footer"), catchword, or similar material appearing on the current page. fw

Attributes:

type classifies the material encoded according to some useful typology.
Data type: CDATA
Sample values include:

header a running title at the top of the page
footer a running title at the bottom of the page
pag a page number or foliation symbol

sig a signature or gathering symbol
catch a catch-word
Default: #IMPLIED

place indicates where on the page this material appears.
Data type: CDATA
Sample values include:

top top of the page.
bot bottom of the page.
left in left margin.
right in right margin.

Default: #IMPLIED

Example:

```
<fw type=sig place=bot>C3</fw>
```

Where running heads are consistent throughout a chapter or section, it is usually more convenient to relate them to the chapter or section, e.g. by use of the rend attribute. The <fw> element is intended for cases where the running head changes from page to page, or where details of page layout and the internal structure of the running heads, are of paramount importance.
Part: additional tag set for transcription of primary sources
Member of classes:
DTD file: teitran2
Data description: May contain character data and phrase-level elements.
Occurs within: [none]
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT fw            - O  (%phrase.seq;)                    >

<!ATTLIST fw                 %a.global
          type               CDATA              #IMPLIED
          place              CDATA              #IMPLIED       >
```

Discussed in: 18.3 ('Headers, Footers, and Similar Matter') on p. 465

gap

<gap> (i.e. omitted material) indicates a point where material has been omitted in a transcription, whether for editorial reasons described in the TEI header, as part of sampling practice, or because the material is illegible or inaudible.

Attributes:

desc (i.e. description) gives a description of the omitted text.
Data type: CDATA
Value: a prose description of the material omitted.
Default: #IMPLIED

reason gives the reason for omission. Sample values include "sampling", "illegible", "inaudible", "irrelevant", "canceled", "canceled and illegible".
Data type: CDATA
Value: any short indication of the reason for the omission.
Default: #IMPLIED

resp (i.e. responsibility) indicates the editor, transcriber or encoder responsible for the decision not to provide any transcription of the text and hence the application of the <gap> tag.
Data type: IDREF
Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).
Default: %INHERITED

hand In the case of text omitted from the transcription because of deliberate deletion by an identifiable hand, signifies the hand which made the deletion.
Data type: IDREF
Value: must be one of the hand identifiers declared in the document header (see section 18.2.1 ('Document Hands') on p. 456).
Default: %INHERITED

agent In the case of text omitted from the transcription because of damage or other phenomenon resulting from an identifiable cause, signifies the causative agent.
Data type: CDATA
Value: any prose description of the agency of damage.
Default: #IMPLIED

extent indicates approximately how much text has been omitted from the transcription, in letters, minims, inches, or any appropriate unit, either because of editorial policy or because a deletion, damage or other cause has rendered transcription impossible.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED
It is possible, but not always necessary, to provide measurements precise to the millimeter or even to the printer's point. The degree of precision attempted will vary with the purpose of the encoding and the nature of the material.

The <gap>, <unclear>, and <del> core tag elements may be closely allied in use with the <damage> and <supplied> elements, available when using the additional tagset for transcription of primary sources. See section 18.2.4 ('The Use of the Gap, Del, Damage, Unclear and Supplied Tags in Combination') on p. 462 for discussion of which element is appropriate for which circumstance.

Part: additional tag set for common core features
Member of classes: edit [and indirectly also:] phrase
DTD file: teicore2
Data description: Empty.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: [none]
Declaration:

```
<!ELEMENT gap           - O  EMPTY                              >

<!ATTLIST gap                %a.global
          desc               CDATA                #IMPLIED
          reason             CDATA                #IMPLIED
          resp               IDREF                %INHERITED

          hand               IDREF                %INHERITED

          agent              CDATA                #IMPLIED
          extent             CDATA                #IMPLIED      >
```

Discussed in: 6.5.3 ('Additions, Deletions and Omissions') on p. 144

gen

<gen> (i.e. gender) identifies the morphological gender of a lexical item, as given in the dictionary.

Attributes: [None: global and inherited attributes only.]
This element is synonymous with <gram type=gender>.
Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, morphInfo
DTD file: teidict2
Data description: May contain character data and phrase-level elements. Typical content will be "m", "f", "n" etc.
Occurs within: etym trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT gen           - -  (%paraContent)                >
<!ATTLIST gen                %a.global
                             %a.dictionaries               >
```

Discussed in: 12.3.1 ('Information on Written and Spoken Forms') on p. 279

genName

<genName> contains a name component used to indicating generational information, such as "Junior", or a number used in a monarch's name.

Attributes: [None: global and inherited attributes only.]
Example:

```
<persName>
  <foreName>Charles</foreName>
  <genName>II</genName>
</persName>
```

Part: additional tag set for names and dates
Member of classes: personPart [and indirectly also:] names
DTD file: teind2
Occurs within: persName
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT genName       - -  (%phrase.seq;)                >
<!ATTLIST genName            %a.global
                             %a.personPart                 >
```

Discussed in: 20.1 ('Personal Names') on p. 488

geog

<geog> (i.e. geographical feature name) contains a common noun identifying some geographical feature contained within a geographic name, such as "valley", "mount" etc.

Attributes: [None: global and inherited attributes only.]
Example:

```
<geogName>
  The <geog>vale</geog>
  of White Horse
</geogName>
```

Part: auxiliary tag set for names and dates
Member of classes: placePart [and indirectly also:] names
DTD file: teind2
Occurs within: geogName placeName
May contain: #PCDATA
Declaration:

```
<!ELEMENT geog          - -  (#PCDATA)                    >
<!ATTLIST geog               %a.global
                             %a.placePart                 >
```

Discussed in: 20.2 ('Place Names') on p. 493

<geogName> (i.e. geographical name) a name associated with some geographical feature such as "Windrush Valley" or "Mount Sinai". geogName

Attributes: [None: global and inherited attributes only.]
Example:

```
<geogName>
   <geog>Mount</geog>
   <name>Sinai</name>
</geogName>
```

Part: additional tag set for names and dates
Member of classes: placePart [and indirectly also:] names
DTD file: teind2
Occurs within: placeName
May contain: #PCDATA geog name
Declaration:

```
<!ELEMENT geogName      - -  (geog | name | #PCDATA)*     >
<!ATTLIST geogName           %a.global
                             %a.placePart                 >
```

Discussed in: 20.2 ('Place Names') on p. 493

<gi> (i.e. generic identifier) contains the name (generic identifier) of an SGML element. gi

Attributes:

TEI indicates whether this element is part of the TEI encoding scheme (i.e., defined in a TEI DTD fragment) or not.
Data type: (YES | NO)
Sample values include:

yes this element is part of the TEI scheme.
no this element is not part of the TEI scheme.

Default: YES

Example:

```
A  <gi tei=n>blort</gi> stalked the prairies.
The element <gi>sourceDesc</gi> is missing.
```

Part: auxiliary tag set for tag set declarations
Member of classes: sgmlKeywords [and indirectly also:] phrase
DTD file: teitsd2
Data description: TEI naming rules require lower case letters for all identifiers, except for names derived from multi-word phrases, in which case all but the first word is capitalized.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagDoc tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: #PCDATA
Declaration:

```
<!ELEMENT gi            - O  (#PCDATA)                          >

<!ATTLIST gi                 %a.global
          TEI                (yes | no)          yes             >
```

Discussed in: 27 ('Tag Set Documentation') on p. 601; 27.1 ('The TagDoc Documentation Element') on p. 603

gloss

<gloss> identifies a phrase or word used to provide a gloss or definition for some other word or phrase.

Attributes:

target identifies the associated term element
Data type: IDREF
Value: must be a valid identifier for some <term> element in the current document
Default: #IMPLIED

Example:

```
We may define <term rend=sc id=tdpv>discoursal point of view</term>
as <gloss target=tdpv>the relationship, expressed through discourse
structure, between the implied author or some other addresser,
and the fiction.</gloss>
```

Part: base tag set for common core features
Member of classes: hqphrase [and indirectly also:] phrase
DTD file: teicore2
Data description: free prose
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT gloss         - -  (%phrase.seq;)                     >

<!ATTLIST gloss              %a.global
          target             IDREF               #IMPLIED      >
```

Discussed in: 6.3.4 ('Terms, Glosses, and Cited Words') on p. 130

gram

<gram> (i.e. grammatical information) within an entry in a dictionary or a terminological data file, contains grammatical information relating to a term, word, or form.

Attributes:

type classifies the grammatical information given according to some convenient typology — in the case of terminological information, preferably the dictionary of data element types specified in ISO WD 12 620.
Data type: CDATA
Sample values include:

pos part of speech (any of the word classes to which a word may be assigned in a given language, based on form, meaning, or a combination of features, e.g. noun, verb, adjective, etc.)

gen gender (formal classification by which nouns and pronouns, and often accompanying modifiers, are grouped and inflected, or changed in form, so as to control certain syntactic relationships)

num number (e.g. singular, plural, dual, ...)

animate animate or inanimate

proper proper noun or common noun

Default: #IMPLIED

A much fuller list of values for the type attribute may be generated from the dictionary of data element types under preparation as ISO TC 37/SC 3/WD 12 620, Computational Aids in Terminology. See ISO 12 620 for fuller details.

In terminological data, the <gram> element usually refers to the most recently specified <term> or <otherForm> element. In flat term entries, the group and depend attributes may be used to indicate exceptions to this general rule. In dictionaries, the element typically relates to the form or forms with which it is grouped in a <form> or other grouping element.

Part: base tag set for terminological data

Member of classes: dictionaries, morphInfo

DTD file: teidict2 teite2n teite2f

Data description: May contain character data and phrase-level elements.

Occurs within: etym ofig termEntry tig

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref

Declaration:

```
<!ELEMENT gram          - O  (%paraContent;)                  >
<!ATTLIST gram               %a.global
                             %a.dictionaries
          type               CDATA               #IMPLIED    >
```

Discussed in: 13.4.2 ('DTD Fragment for Flat Style') on p. 322; 13.2 ('Tags for Terminological Data') on p. 312; 12.3.2 ('Grammatical Information') on p. 284

<gramGrp> (i.e. grammatical information group) groups morpho-syntactic information about a lexical item, e.g. <pos>, <gen>, <number>, <case>, or <itype> (inflectional class). gramGrp

Attributes: [None: global and inherited attributes only.]

Part: base tag set for printed dictionaries

Member of classes: dictionaries, dictionaryParts, dictionaryTopLevel, gramInfo

DTD file: teidict2

Data description: May contain character data mixed with any other elements defined in the dictionary tag set.

Occurs within: entry gramGrp hom re sense trans

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl colloc corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss gramGrp handShift hi label lang lbl link list listBibl m measure mentioned move name note num orig oRef oVar phr pos ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage subc table tag tech term text time timeRange timeStruct title usg val view w xptr xref

Declaration:

```
<!ELEMENT gramGrp       - -  (%m.gramInfo | %paraContent)*    >
<!ATTLIST gramGrp            %a.global
                             %a.dictionaries                  >
```

Discussed in: 12.3.2 ('Grammatical Information') on p. 284

<graph> encodes a graph, which is a collection of nodes, and arcs which connect the nodes. graph

Attributes:

type describes the type of graph.
Data type: CDATA
Sample values include:

undirected undirected graph

directed directed graph

transition network a directed graph with distinguished initial and final nodes

transducer a transition network with up to two labels on each arc

Default: #IMPLIED
If type=undirected, then the distinction between the to and from attributes of the <arc> tag is neutralized. Also, the adj attribute, rather than the adjFrom and adjTo attributes, should be used to encode pointers to the ends of the arcs. If type=directed (or any other value which implies directionality), then the adjFrom and adjTo attributes should be used, instead of the adj attribute.

label gives a label for a graph.
Data type: CDATA
Value: A character string.
Default: #IMPLIED

order states the order of the graph, i.e., the number of its nodes.
Data type: NUMBER
Value: A positive integer.
Default: #IMPLIED

size states the size of the graph, i.e., the number of its arcs.
Data type: NUMBER
Value: A nonnegative integer.
Default: #IMPLIED

Example:

```
<graph type=undirected
       id=cug1
       label='Airline Connections in Southwestern USA'
       rend='LABEL-PLACE bottom center NODE-FRAME none ARC solid line'
       order=5
       size=4>
  <node label=LAX id=LAX degree=2>
  <node label=LVG id=LVG degree=2>
  <node label=PHX id=PHX degree=3>
  <node label=TUS id=TUS degree=1>
  <node label=CIB id=CIB degree=0>
  <arc  from=LAX to=LVG>
  <arc  from=LAX to=PHX>
  <arc  from=LVG to=PHX>
  <arc  from=PHX to=TUS>
  </graph>
```

Example:

```
<graph type=directed
       id=rdg2
       label='Selected Airline Routes in Southwestern USA'
       rend='LABEL-PLACE bottom center NODE-FRAME none ARC solid line
with arrowhead'
       order=5
       size=5>
   <node label=LAX id=LAX inDegree=1 outDegree=1 adjTo=LVG adjFrom=PHX>
   <node label=LVG id=LVG inDegree=1 outDegree=1 adjFrom=LAX adjTo=PHX>
   <node label=PHX id=PHX inDegree=2 outDegree=2 adjTo='LAX TUS'
         adjFrom='LVG TUS'>
   <node label=TUS id=TUS inDegree=1 outDegree=1 adjTo=PHX adjFrom=PHX>
```

```
<node label=CIB id=CIB inDegree=0 outDegree=0>
</graph>
```

Part: additional tag set for graphs, networks, and trees
Member of classes: chunk [and indirectly also:] common
DTD file: teinet2
Data description: One or more nodes and zero or more arcs in any order.
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage view
May contain: arc node
Declaration:

```
<!ELEMENT graph         - -  ((node)+ & (arc)*)                >
<!ATTLIST graph              %a.global
          type               CDATA               #IMPLIED
          label              CDATA               #IMPLIED
          order              NUMBER              #IMPLIED
          size               NUMBER              #IMPLIED       >
```

Discussed in: 21.1 ('Graphs and Digraphs') on p. 506

group

<group> contains the body of a composite text, grouping together a sequence of distinct texts (or groups of such texts) which are regarded as a unit for some purpose, for example the collected works of an author, a sequence of prose essays, etc.

Attributes: [None: global and inherited attributes only.]
Part: base tag set for common core features
Member of classes: declaring
DTD file: teistr2
Occurs within: group text
May contain: argument byline closer docAuthor docDate epigraph group head opener salute signed text trailer
Declaration:

```
<!ELEMENT group         - O  ((%m.divtop;)*, (text | group)+,
                              (%m.divbot;)*)                  >
<!ATTLIST group              %a.global
                             %a.declaring                     >
```

Discussed in: 7 ('Default Text Structure') on p. 183; 7.3 ('Groups of Texts') on p. 195; 23.1 ('Varieties of Composite Text') on p. 538

hand

<hand> used in the header to define each distinct scribe or handwriting style.

Attributes:

hand unique identifier, either numeric or alphanumeric, used thereafter in the document to refer to this scribe or handwriting style.
Data type: CDATA
Default: #REQUIRED

scribe gives the name of, or other identifier for, the scribe.
Data type: CDATA
Value: 'tremulous hand', 'hand b', 'Hoccleve'
Default: #IMPLIED

style indicates recognized writing styles.
Data type: CDATA
Value: 'secretary', 'copperplate', 'Chancery', 'Italian', etc.
Default: #IMPLIED

lang (i.e. language) indicates dominant language of hand.
Data type: CDATA
Value: 'Latin', 'English'
Default: #IMPLIED

ink describes colour of ink, e.g. 'brown'. May also be used to indicate the writing medium, e.g. 'pencil',

Data type: CDATA
Default: #IMPLIED

character used to describe other characteristics of the hand, particularly those related to the quality of the writing.
Data type: CDATA
Value: 'shaky', 'thick', 'regular'
Default: #IMPLIED

first indicates the first scribe in the document.
Data type: CDATA
Value: 'YES', 'NO'
Default: #IMPLIED

resp (i.e. responsible) signifies the editor or transcriber responsible for identifying the hand.
Data type: CDATA
Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).
Default: %INHERITED

The <hand> element is used in the header to define each unique hand, including unique scribes, distinguished by the encoder in the document. One such element must appear within the header for each hand distinguished in the text. Each location where a change of hands occurs is marked in the text by the <handShift> milestone element.

Part: additional tag set for transcription of primary sources
Member of classes:
DTD file: teitran2
Data description: Empty.
Occurs within: handList
May contain: [none]
Declaration:

```
<!ELEMENT hand          - O  EMPTY                              >

<!ATTLIST hand               %a.analysis
                             %a.linking
                             %a.terminology
          id                 ID                  #IMPLIED
          n                  CDATA               #IMPLIED
          rend               CDATA               #IMPLIED
          hand               CDATA               #REQUIRED
          scribe             CDATA               #IMPLIED
          style              CDATA               #IMPLIED
          lang               CDATA               #IMPLIED
          ink                CDATA               #IMPLIED
          character          CDATA               #IMPLIED
          first              CDATA               #IMPLIED
          resp               CDATA               %INHERITED     >
```

Discussed in: 18.2.1 ('Document Hands') on p. 456

handList

<handList> contains a series of <hand> elements listing the different hands of the source.

Attributes: [None: global and inherited attributes only.]
Part: additional tag set for transcription of primary sources
Member of classes:
DTD file: teitran2
Data description: Contains a series of <hand> elements.
Occurs within: profileDesc
May contain: hand
Declaration:

```
<!ELEMENT handList      - O  (hand*)                            >
```

```
<!ATTLIST handList          %a.global                       >
```

Discussed in: 18.2.1 ('Document Hands') on p. 456

handShift

<handShift> marks the beginning of a sequence of text written in a new hand, or of a change in the scribe, writing style, ink or character of the document hand.

Attributes:

new identifies the new hand.
Data type: IDREF
Value: must be one of the hand identifiers declared in the document header (see section 18.2.1 ('Document Hands') on p. 456).
Default: #IMPLIED

old identifies the old hand.
Data type: IDREF
Value: must be one of the hand identifiers declared in the document header (see section 18.2.1 ('Document Hands') on p. 456).
Default: #IMPLIED

style indicates recognised writing styles
Data type: CDATA
Value: 'secretary', 'copperplate', 'Chancery', 'Italian', etc.
Default: #IMPLIED

ink describes colour of ink, e.g. 'brown'. May also be used to indicate the writing medium, e.g. 'pencil'
Data type: CDATA
Default: #IMPLIED

character used to describe other characteristics of the hand, particularly those related to the quality of the writing.
Data type: CDATA
Value: 'shaky', 'thick', 'regular'
Default: #IMPLIED

resp (i.e. responsible) signifies the editor or transcriber responsible for identifying the change of hand.
Data type: IDREF
Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).
Default: %INHERITED

The <handShift> element may be used either to denote a shift in the document hand (as from one scribe to another, on one writing style to another). Or, it may indicate a shift within a document hand, as a change of writing style, character or ink

Part: additional tag set for transcription of primary sources
Member of classes: phrase
DTD file: teitran2
Data description: Empty.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: [none]
Declaration:

```
<!ELEMENT handShift     - O  EMPTY                          >

<!ATTLIST handShift          %a.global
          new                IDREF              #IMPLIED
          old                IDREF              #IMPLIED
          style              CDATA              #IMPLIED
          ink                CDATA              #IMPLIED
          character          CDATA              #IMPLIED
          resp               IDREF              %INHERITED    >
```

Discussed in: 18.2.1 ('Document Hands') on p. 456

head

<head> (i.e. heading) contains any heading, for example, the title of a section, or the heading of a list or glossary.

Attributes:

type categorizes the heading in some way meaningful to the encoder.
Data type: CDATA
Value: A set of user-defined keywords may be employed. Their significance should be documented in the header.
Default: #IMPLIED

The most common use for the <head> element is to mark the headings of sections. In older writings, the headings or *incipits* may be rather longer than usual in modern works. If a section has an explicit ending as well as a heading, it should be marked as a <trailer>, as in this example:

Example:

```
<div1 name=book n='I'><head>In the name of Christ here begins
the first book of the ecclesiastical history of Georgius
Florentinus, known as Gregory, Bishop of Tours.</head>
<div2><head>Chapter-Headings</head>
<!-- list of chapter heads omitted ...  -->
<div2><head>In the name of Christ here begins Book I of the
history.</head>
<p>Proposing as I do ...
<p>From the Passion of our Lord until the death of Saint Martin
four hundred and twelve years passed.
<trailer>Here ends the first Book, which covers five thousand,
five hundred and ninety-six years from the beginning of the
world down to the death of Saint Martin.</trailer>
</div2>
</div1>
```

The <head> tag is also used to mark headings of other units, such as lists:

Example:

```
With a few exceptions, connectives are equally useful in
all kinds of discourse:  description, narration, exposition,
argument.
<list type=simple>
<head>Connectives</head>
<item>above
<item>accordingly
<item>across from
<item>adjacent to
<item>again
<item> ...
</list>
```

The <head> tag is used for headings at all levels; processing programs which treat (e.g.) chapter headings, section headings, and list titles differently must determine the proper processing of a <head> element based on its structural position. A <head> occurring as the first element

of a list is the title of that list; one occurring as the first element of a <div1> is the title of that chapter or section.
Part: additional tag set for common core features
Member of classes: divtop
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: argument back body castGroup castList div div0 div1 div2 div3 div4 div5 div6 div7 epilogue figure front group lg lg1 lg2 lg3 lg4 lg5 list listBibl performance prologue set table
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT head          - O  (%paraContent;)                  >

<!ATTLIST head               %a.global
          type               CDATA                #IMPLIED      >
```

Discussed in: 6.7 (‘Lists’) on p. 149; 7.2 (‘Elements Common to All Divisions’) on p. 190

headItem

<headItem> (i.e. heading for list items) contains the heading for the item or gloss column in a glossary list or similar structured list.

Attributes: [None: global and inherited attributes only.]
Example:

```
The simple, straightforward statement of an idea is
preferable to the use of a worn-out expression.
<list type=gloss>
<head.label rend='small caps'>TRITE
<headItem  rend='small caps'>SIMPLE, STRAIGHTFORWARD
<label>bury the hatchet  <item>stop fighting, make peace
<label>at loose ends     <item>disorganized
<label>on speaking terms <item>friendly
<label>fair and square   <item>completely honest
<label>at death's door   <item>near death
</list>
```

The <headItem> element may appear only if each item in the list is preceded by a <label>.
Part: additional tag set for common core features
Member of classes:
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: list
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT headItem      - O  (%phrase.seq;)                   >

<!ATTLIST headItem           %a.global                        >
```

Discussed in: 6.7 (‘Lists’) on p. 149

headLabel

<headLabel> (i.e. heading for list labels) contains the heading for the label or term column in a glossary list or similar structured list.

Attributes: [None: global and inherited attributes only.]
Example:

```
The simple, straightforward statement of an idea is
preferable to the use of a worn-out expression.
<list type=gloss>
<headLabel rend='small caps'>TRITE
```

```
<head.item  rend='small caps'>SIMPLE, STRAIGHTFORWARD
<label>bury the hatchet  <item>stop fighting, make peace
<label>at loose ends         <item>disorganized
<label>on speaking terms <item>friendly
<label>fair and square         <item>completely honest
<label>at death's door   <item>near death
</list>
```

The <headLabel> element may appear only if each item in the list is preceded by a <label>.

Part: additional tag set for common core features
Member of classes:
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: list
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT headLabel     - O  (%phrase.seq;)                    >

<!ATTLIST headLabel          %a.global                         >
```

Discussed in: 6.7 ('Lists') on p. 149

hi

<hi> (i.e. highlighted) marks a word or phrase as graphically distinct from the surrounding text, for reasons concerning which no claim is made.

Attributes:

rend (i.e. rendition) describes the rendition or presentation of the word or phrase highlighted.
Data type: CDATA
Value: a set of keywords from some suitable vocabulary; no systematic recommendations for such a vocabulary are made by these Guidelines.
Default: #IMPLIED
The rend attribute, though optional in general, is recommended on the <hi> element.

Example:

```
<hi rend=gothic>And this Indenture further witnesseth</hi>
that the said <hi rend=italic>Walter Shandy</hi>, merchant,
in consideration of the said intended marriage ...
```

Part: auxiliary tag set for TEI headers
Member of classes: hqphrase [and indirectly also:] phrase
DTD file: teicore2
Data description: free prose
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label

lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref

Declaration:

```
<!ELEMENT hi            - -  (%paraContent;)                    >

<!ATTLIST hi                 %a.analysis
                             %a.linking
                             %a.terminology
          id                 ID                  #IMPLIED
          n                  CDATA               #IMPLIED
          lang               IDREF               %INHERITED

          rend               CDATA               #IMPLIED      >
```

Discussed in: 6.3.2.2 ('Emphatic Words and Phrases') on p. 125; 6.3.2 ('Emphasis, Foreign Words, and Unusual Language') on p. 124

<hom> (i.e. homograph) groups information relating to one homograph within an <entry>. — hom

Attributes: [None: global and inherited attributes only.]
Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts
DTD file: teidict2
Data description: May contain character data mixed with any other elements defined in the dictionary tag set.
Occurs within: entry trans
May contain: def eg etym form gramGrp note re sense trans usg xr [May not include at any level: entry]
Declaration:

```
<!ELEMENT hom           - O  (sense | %m.dictionaryTopLevel)*
                                              -(entry)      >

<!ATTLIST hom                %a.global
                             %a.dictionaries                >
```

Discussed in: 12.2 ('The Structure of Dictionary Entries') on p. 274

<hour> (i.e. hour) the hour component of a temporal expression such as — hour

Attributes: [None: global and inherited attributes only.]
Example:

```
At the third stroke the time sponsored by Accurist
will be
<timeStruct  value='1821:30'>
  <hour>six</hour>
  <minute>twenty-one</minute and
  <second>thirty</second> seconds
</timeStruct>
```

Part: additional tag set for names and dates
Member of classes: temporalExpr
DTD file: teind2
Occurs within: dateStruct timeStruct
May contain: #PCDATA
Declaration:

```
<!ELEMENT hour          - -  (#PCDATA)                          >

<!ATTLIST hour               %a.global
                             %a.temporalExpr                >
```

Discussed in: 20.4 ('Dates and Time') on p. 499

<hyph> (i.e. hyphenation) contains a hyphenated form of a dictionary headword, or hyphenation information in some other form. — hyph

Attributes: [None: global and inherited attributes only.]

Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, formInfo
DTD file: teidict2
Data description: May contain character data and phrase-level elements.
Occurs within: form trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT hyph          - O  (%paraContent;)                    >
<!ATTLIST hyph               %a.global
                             %a.dictionaries                    >
```

Discussed in: 12.3.1 ('Information on Written and Spoken Forms') on p. 279

hyphenation

<hyphenation> (i.e. Hyphenation) summarizes the way in which hyphenation in a source text has been treated in an encoded version of it.

Attributes:

eol indicates whether or not end-of-line hyphenation has been retained in a text.
Data type: (ALL | SOME | NONE)
Sample values include:

all all end-of-line hyphenation has been retained, even though the lineation of the original may not have been.

some end-of-line hyphenation has been retained in some cases.

hard all 'soft' end-of-line hyphenation has been removed: any remaining hyphenation represents 'hard hyphens'.

none all end-of-line hyphenation has been removed: any remaining hyphenation occurred within the line.

Default: SOME

Example:

```
<hyphenation eol=some>
<p>End-of-line hyphenation silently removed where appropriate
```

Part: auxiliary tag set for TEI headers
Member of classes: declarable
DTD file: teihdr2
Data description: prose
Occurs within: editorialDecl
May contain: p
Declaration:

```
<!ELEMENT hyphenation   - O  (p+)                               >
<!ATTLIST hyphenation        %a.global
                             %a.declarable
          eol                (all | some | none) some           >
```

Discussed in: 5.3.3 ('The Editorial Practices Declaration') on p. 96; 23.3.2 ('Declarable Elements') on p. 552

idno

<idno> (i.e. identifying number) supplies any standard or non-standard number used to identify a bibliographic item.

Attributes:

type categorizes the number, for example as an ISBN or other standard series.
Data type: CDATA
Value: A name or abbreviation indicating what type of identifying number is given (e.g. ISBN, LCCN).
Default: #IMPLIED

Example:

```
<idno type=ISSN>0143-3385
<idno type=OTA>116
```

Part: auxiliary tag set for TEI headers
Member of classes: biblPart
DTD file: teihdr2
Data description:
Occurs within: bibl biblStruct publicationStmt seriesStmt
May contain: #PCDATA
Declaration:

```
<!ELEMENT idno          - O  (#PCDATA)                          >

<!ATTLIST idno               %a.global
          type               CDATA                #IMPLIED       >
```

Discussed in: 5.2.4 ('Publication, Distribution, etc') on p. 86; 5.2.5 ('The Series Statement') on p. 88; 6.10.2.3 ('Imprint, Pagination, and Other Details') on p. 171

<if> defines a conditional default value for a feature; the condition is specified as a feature structure, and is met if it *subsumes* the feature structure in the text for which a default value is sought. if

Attributes: [None: global and inherited attributes only.]
Part: auxiliary tag set for feature system declarations
Member of classes:
DTD file: teifsd2
Data description: May contain a feature structure, followed by a feature value; the two are separated by a <then> element.
Occurs within: vDefault
May contain: alt any dft f fs fAlt minus msr nbr none null plus rate str sym then uncertain vAlt
Declaration:

```
<!ELEMENT if            - -  ((fs | f | fAlt), then,
                              (%m.featureVal) )                  >

<!ATTLIST if                 %a.global                           >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

<iff> separates the condition from the consequence in a <bicond> element. iff

Attributes: [None: global and inherited attributes only.]
This element is provided primarily to enhance the human readability of the feature-system declaration.
Part: auxiliary tag set for feature system declarations
Member of classes:
DTD file: teifsd2
Data description: Empty.
Occurs within: bicond
May contain: [none]
Declaration:

```
<!ELEMENT iff           - O  EMPTY                              >

<!ATTLIST iff                %a.global                          >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

<ihs> (i.e. independent header set) contains a set of TEI Headers exchanged independently of their associated text or group elements. ihs

Attributes: [None: global and inherited attributes only.]
Example:

```
<ihs>
<teiHeader> ... </teiHeader>
<teiHeader> ... </teiHeader>
<teiHeader> ... </teiHeader>
<!-- ... -->
</ihs>
```

This element is required.
Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teishd2
Data description: Contains an arbitrary sequence of <teiHeader> elements only.
Occurs within: [none]
May contain: teiHeader
Declaration:

```
<!ELEMENT ihs           - O  (teiHeader+)                >
<!ATTLIST ihs                %a.global                   >
```

Discussed in: 24.8 ('Structure of the DTD for Independent Headers') on p. 568

imprimatur

<imprimatur> contains a formal statement authorizing the publication of a work, sometimes required to appear on a title page or its verso.

Attributes: [None: global and inherited attributes only.]
Example:

```
<imprimatur>Licensed and entred acording to Order.
```

Part: base tag set for common core features
Member of classes: tpParts
DTD file: teifron2
Data description:
Occurs within: titlePage
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT imprimatur    - O  (%paraContent;)             >
<!ATTLIST imprimatur         %a.global                   >
```

Discussed in: 7.5 ('Title Pages') on p. 203

imprint

<imprint> groups information relating to the publication or distribution of a bibliographic item.

Attributes: [None: global and inherited attributes only.]
Example:

```
<imprint>Oxford, Clarendon Press, 1987
</imprint>
<imprint><pubPlace>Oxford</pubPlace>
        <publisher>Clarendon Press</publisher>
        <date>1987</date>
</imprint>
```

Part: base tag set for common core features
Member of classes: biblPart
DTD file: teicore2
Data description:
Occurs within: bibl monogr
May contain: biblScope date publisher pubPlace
Declaration:

```
<!ELEMENT imprint       - O  (pubPlace | publisher | date |
                              biblScope)*                    >

<!ATTLIST imprint            %a.global                       >
```

Discussed in: 6.10.2.3 ('Imprint, Pagination, and Other Details') on p. 171

<index> (i.e. index entry) marks a location to be indexed for whatever purpose. index

Attributes:

index (i.e. index number) indicates which index (of several) the index entry belongs to.
Data type: CDATA
Value: any string of characters; valid values are application-dependent.
Default: #IMPLIED
This attribute makes it possible to create multiple indices for a text.

level1 (i.e. first-level index entry) gives the form under which the index entry is to be made.
Data type: CDATA
Value: any string of characters.
Default: #REQUIRED
At least one level of entry is required.

level2 (i.e. second-level index entry) gives the second-level form, if any.
Data type: CDATA
Value: any string of characters.
Default: #IMPLIED

level3 (i.e. third-level index entry) gives the third-level form, if any.
Data type: CDATA
Value: any string of characters.
Default: #IMPLIED

level4 (i.e. fourth-level index entry) gives the fourth-level form, if any.
Data type: CDATA
Value: any string of characters.
Default: #IMPLIED

Example:

```
David's other principal backer, Josiah ha-Kohen
<index level1='Josiah ha-Kohen b. Azarya'
       level2='cousin and backer of David b. Daniel'>
b. Azarya, son of one of the last gaons of Sura,
<index level1='Azarya' level2='gaon of Sura'>
<index level1='Sura'>
was David's own first cousin.
```

Part: additional tag set for common core features
Member of classes:
DTD file: teicore2
Data description: Empty.
Occurs within: [none]
May contain: [none]
Declaration:

```
<!ELEMENT index         - O  EMPTY                          >

<!ATTLIST index              %a.global
          index              CDATA              #IMPLIED
          level1             CDATA              #REQUIRED
          level2             CDATA              #IMPLIED
          level3             CDATA              #IMPLIED
          level4             CDATA              #IMPLIED    >
```

Discussed in: 6.8.2 ('Index Entries') on p. 154

<iNode> (i.e. intermediate (or internal) node) represents an intermediate (or internal) node of a tree. iNode

Attributes:

label gives a label for an intermediate node.
Data type: CDATA
Value: A character string.
Default: #IMPLIED

value provides the value of an intermediate node, which is a feature structure or other analytic element.
Data type: IDREF
Value: A valid identifier of a feature structure or other analytic element.
Default: #IMPLIED

children provides a list of IDs of the elements which are the children of the intermediate node.
Data type: IDREFS
Value: A list of IDs.
Default: #REQUIRED

parent provides the ID of the element which is the parent of this node.
Data type: IDREF
Value: The ID of the parent node.
Default: #IMPLIED

ord indicates whether or not the internal node is ordered.
Data type: (Y | N)
Sample values include:

Y indicates that the children of the intermediate node are ordered.

N indicates that the children of the intermediate node are unordered.

Default: #IMPLIED
Use if and only if ord=partial is specified on the <tree> tag and the intermediate node has more than one child.

follow provides an ID of the element which this node follows.
Data type: IDREF
Value: An ID of another intermediate node or leaf of the tree.
Default: #IMPLIED
If the tree is unordered or partially ordered, this attribute has the property of fixing the relative order of the intermediate node and the element which is the value of the attribute.

outDegree gives the out degree of an intermediate node, the number of its children.
Data type: NUMBER
Value: A nonnegative integer.
Default: #IMPLIED
The in degree of an intermediate node is always 1.

Example:

```
<iNode label=PT id=PT1 parent=VB1 children=up1 outDegree=1 follow=PN1>
```

Part: additional tag set for graphs, networks, and trees
Member of classes:
DTD file: teinet2
Data description: empty
Occurs within: tree
May contain: [none]
Declaration:

```
<!ELEMENT iNode         - O  EMPTY                              >

<!ATTLIST iNode              %a.global
          label              CDATA               #IMPLIED
          value              IDREF               #IMPLIED
          children           IDREFS              #REQUIRED
          parent             IDREF               #IMPLIED
          ord                (Y | N)             #IMPLIED
```

```
follow          IDREF           #IMPLIED
outDegree       NUMBER          #IMPLIED      >
```

Discussed in: 21.2 ('Trees') on p. 514

interaction

<interaction> describes the extent, cardinality and nature of any interaction among those producing and experiencing the text, for example in the form of response or interjection, commentary etc.

Attributes:

type specifies whether or not there is any interaction between active and passive participants in the text.
Data type: (NONE | PARTIAL | COMPLETE | INAPPLICABLE)
Sample values include:
none no interaction of any kind, e.g. a monologue
partial some degree of interaction, e.g. a monologue with set responses
complete complete interaction, e.g. a face to face conversation
inapplicable this parameter is inappropriate or inapplicable in this case
Default: #IMPLIED

active specifies the number of active participants (or *addressors*) producing parts of the text.
Data type: CDATA
Sample values include:
singular a single addressor
plural many addressors
corporate a corporate addressor
unknown number of addressors unknown or unspecifiable
Default: #IMPLIED

passive specifies the number of passive participants (or *addressees*) to whom a text is directed or in whose presence it is created or performed.
Data type: CDATA
Sample values include:
self text is addressed to the originator e.g. a diary
single text is addressed to one other person e.g. a personal letter
many text is addressed to a countable number of others e.g. a conversation in which all participants are identified
group text is addressed to an undefined but fixed number of participants e.g. a lecture
world text is addressed to an undefined and indeterminately large number e.g. a published book
Default: #IMPLIED

Example:

```
<!-- An informal conversation:  ->
<interaction type=complete active=plural passive=many>
```

Example:

```
<!-- A formal lecture:-->
<interaction type=none active=singular passive=group>
```

Part: auxiliary tag set for corpora and collections
Member of classes:
DTD file: teicorp2
Data description:
Occurs within: textDesc
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT interaction   - O  (%phrase.seq;)                    >
```

```
<!ATTLIST interaction        %a.global
          type               (none | partial | complete |
                             inapplicable)       #IMPLIED
          active             CDATA               #IMPLIED
          passive            CDATA               #IMPLIED      >
```

Discussed in: 23.2.1 ('The Text Description') on p. 541

interp

<interp> (i.e. interpretation) provides for an interpretative annotation which can be linked to a span of text.

Attributes:

value identifies the specific phenomenon being annotated.
Data type: CDATA
Value: Any string of characters.
Default: #REQUIRED

Example:

```
<interp resp=TMA type='structural unit' value='aftermath'>
```

Part: additional tag set for
Member of classes: interpret, metadata [and indirectly also:] globincl
DTD file: teiana2
Data description: Empty element.
Occurs within: interpGrp
May contain: [none]
Declaration:

```
<!ELEMENT interp        - O  EMPTY                          >

<!ATTLIST interp             %a.global
                             %a.interpret
          value              CDATA               #REQUIRED      >
```

Discussed in: 15.3 ('Spans and Interpretations') on p. 387

interpGrp

<interpGrp> (i.e. interpretation group) collects together <interp> tags.

Attributes: [None: global and inherited attributes only.]
Example:

```
<interpGrp resp=TMA type='collection of structural units'>
 <interp value='introduction'>
 <interp value='conflict'>
 <interp value='climax'>
 <interp value='revenge'>
 <interp value='reconciliation'>
 <interp value='aftermath'>
 </interpGrp>
```

Part: additional tag set for
Member of classes: interpret, metadata [and indirectly also:] globincl
DTD file: teiana2
Data description: Any number of <interp> elements.
Occurs within: [none]
May contain: interp
Declaration:

```
<!ELEMENT interpGrp     - -  ((interp)*)                    >

<!ATTLIST interpGrp          %a.global
                             %a.interpret                   >
```

Discussed in: 15.3 ('Spans and Interpretations') on p. 387

interpretation

<interpretation> describes the scope of any analytic or interpretive information added to the text in addition to the transcription.

Attributes: [None: global and inherited attributes only.]
Example:

```
<interpretation>
  <p>The part of speech analysis applied throughout section 4 was
  added by hand and has not been checked
```

Part: auxiliary tag set for TEI headers
Member of classes: declarable
DTD file: teihdr2
Data description:
Occurs within: editorialDecl
May contain: p
Declaration:

```
<!ELEMENT interpretation
                        - O  (p+)                              >

<!ATTLIST interpretation     %a.global
                             %a.declarable                     >
```

Discussed in: 5.3.3 ('The Editorial Practices Declaration') on p. 96

<item> contains one component of a list. item

Attributes:

n (i.e. name or number) indicates the number (or letter, or other enumerator) borne by this list item in the copy text.
Data type: CDATA
Value: Whatever string of characters is used to label a list item in the copy text may be used as the value of n, but it is not required that numbering be recorded explicitly.
Default: #IMPLIED
In ordered lists, the n attribute on the <item> element is by definition synonymous with the use of the <label> element to record the enumerator of the list item. In glossary lists, however, the term being defined should be given with the <label> element, not n.

Example:

```
<list type=ordered>
<head>Here begin the chapter headings of Book IV</head>
<item n='4.1'>The death of Queen Clotild.
<item n='4.2'>How King Lothar wanted to appropriate
     one third of the Church revenues.
<item n='4.3'>The wives and children of Lothar.
<item n='4.4'>The Counts of the Bretons.
<item n='4.5'>Saint Gall the Bishop.
<item n='4.6'>The priest Cato.
<item> ...
</list>
```

Part: additional tag set for common core features
Member of classes:
DTD file: teicore2
Data description: May contain simple prose or a sequence of chunks.
Occurs within: change list
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph entry entryFree event expan eTree figure foreign formula gap gi gloss graph handShift hi kinesic l label lang lg lg1 link list listBibl m measure mentioned move name note num orig oRef oVar p pause phr ptr pRef pVar q quote ref reg rs s seg shift sic sound soCalled sp stage superentry table tag tech term termEntry text time timeRange timeStruct title tree u val view vocal w writing xptr xref TEI...end
Declaration:

```
<!ELEMENT item          - O  (%specialPara;)                   >
```

```
<!ATTLIST item              %a.analysis
                            %a.linking
                            %a.terminology
          id                ID                  #IMPLIED
          lang              IDREF               %INHERITED

          rend              CDATA               #IMPLIED
          n                 CDATA               #IMPLIED      >
```

Discussed in: 6.7 ('Lists') on p. 149; 5.5 ('The Revision Description') on p. 112

itype

<itype> (i.e. inflectional class) indicates the inflectional class associated with a lexical item.

Attributes:

type indicates the type of indicator used to specify the inflection class, when it is necessary to distinguish between the usual abbreviated indications (e.g. "inv") and other kinds of indicators, such as special codes referring to conjugation patterns, etc.
Data type: CDATA
Sample values include:
abbrev abbreviated indicator
verb table coded reference to a table of verbs
Default: #IMPLIED
This element is synonymous with <gram type='inflectional type'>.

Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, morphInfo
DTD file: teidict2
Data description: May contain character data and phrase-level elements. Typical content will be "invariant", "n 3" etc.
Occurs within: etym trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT itype          - -  (%paraContent)                  >

<!ATTLIST itype                 %a.global
                                %a.dictionaries
          type                  CDATA               #IMPLIED      >
```

Discussed in: 12.3.1 ('Information on Written and Spoken Forms') on p. 279

join

<join> identifies a possibly fragmented segment of text, by pointing at the possibly discontiguous elements which compose it.

Attributes:

targets specifies the SGML identifiers of the elements or passages to be joined into a virtual element.
Data type: IDREFS
Value: Each value specified must be the same as that specified as value for an ID attribute for some other element in the current SGML document.
Default: #REQUIRED

result specifies the name of an element which this aggregation may be understood to represent.
Data type: NAME
Value: The generic identifier of an element in the current DTD.
Default: %INHERITED

desc (i.e. description) a brief phrase describing the virtual element created by the join.
Data type: CDATA
Value: any string of characters.

Default: %INHERITED

The desc attribute may be used to give a brief description of the virtual element, if its content or function are not obvious.

scope indicates whether the targets to be joined include the entire element indicated (the entire subtree including its root), or just the children of the target (the branches of the subtree).

Data type: (ROOT | BRANCHES)

Sample values include:

root the rooted subtrees indicated by the targets attribute are joined, each subtree become a child of the virtual element created by the <join>

branches the children of the subtrees indicated by the targets attribute become the children of the virtual element (i.e. the roots of the subtrees are discarded)

Default: ROOT

Example:

```
<list><head>Authors</head>
  <item id=uf>Figge, Udo
  <item id=ch>Heibach, Christiane
  <item id=gh>Heyer, Gerhard
  <item id=bp>Philipp, Bettina
  <item id=ms>Samiec, Monika
  <item id=ss>Schierholz, Stefan
</list>
<!-- elsewhere ...  -->
<join result='list' targets='ch bp ss' scope=root
      desc='Heidelberger authors'>
```

In this example, the text contains a list of authors of articles in a journal; the <join> element is used to join into one virtual list the names of the authors who reside in Heidelberg.

targOrder specifies whether or not the order in which components of the join are listed on the targets attribute is significant.

Data type: (Y | N | U)

Sample values include:

Y Yes: the order should be followed when combining the targeted elements.

N No: the order has no significance when combining the targeted elements.

U Unspecified: the order may or may not be significant.

Default: Y

This attribute has a different default value on this element from that which it has for other members of the pointer class.

The following example is discussed in section 14.7 ('Aggregation') on p. 369.

Example:

```
<sp who='Hughie'><p>How does it go?
  <q><l id=X1>da-da-da
     <l id=L2>gets a new frog
     <l>...
  </q></p>
<sp who='Louie'><p><q><l id=L1>When the old pond</l> ...</q></p>
<sp who='Dewey'><p><q>... <l id=L3>It's a new pond.</l></q></p>
<!-- ... -->
<join result=lg targets='L1 L2 L3' scope=root
      desc='haiku'>
<!-- a famous haiku by Bashoo, in an eccentric translation -->
```

The attribute targOrder is specified with a value of "Y" to indicate that the order of the three lines is significant. The attribute scope is allowed to take the default value of "root" to indicate that the virtual element being identified is a line group (<lg>) which contains the three <l> elements L1, L2, and L3, and not just their character data content.

In this example, the attribute scope is specified with the value of "branches" to indicate that the virtual list being constructed is to be made by taking the lists indicated by the targets attribute of the <join> element, discarding the <list> tags which enclose them, and combining the items contained within the lists into a single virtual list.

Example:

```
<s>Southern dialect (my own variety, at least) has only
<list id=lp1>
<item><s>I done gone</s></item>
<item><s>I done went</s></item>
</list>
whereas Negro Non-Standard basilect has both these and
<list id=lp2>
<item><s>I done go</s></item>.
</list></s>
<s>White Southern dialect also has
<list id=lp3>
<item><s>I've done gone</s></item>
<item><s>I've done went</s></item>
</list>
which, when they occur in Negro dialect, should probably be
considered as borrowings from other varieties of English.</s>
<!-- ... -->
<join result=list id=lst1 parts='lp1 lp2 lp3' scope=branches
      desc='Sample sentences in Southern speech'>
```

Part: additional tag set for
Member of classes: pointer
DTD file: teilink2.dtd
Data description: Empty element.
Occurs within: joinGrp
May contain: [none]
Declaration:

```
<!ELEMENT join          - O  EMPTY                              >

<!ATTLIST join               %a.global
          type               CDATA               #IMPLIED
          resp               CDATA               #IMPLIED
          crdate             CDATA               #IMPLIED
          targType           NAMES               #IMPLIED
          evaluate           (all | one | none)  #IMPLIED
          targets            IDREFS              #REQUIRED
          result             NAME                %INHERITED

          desc               CDATA               %INHERITED

          scope              (root | branches)   root
          targOrder          (Y | N | U)         Y           >
```

Discussed in: 14.7 ('Aggregation') on p. 369

joinGrp

<joinGrp> (i.e. join group) groups a collection of <join> elements and possibly pointers.

Attributes:

result describes the result of the joins gathered in this collection.
Data type: CDATA
Value: where specified on a <joinGrp> element, it supplies the default value for the result on each <join> included within the group.
Default: #IMPLIED

desc (i.e. description) a brief phrase describing the virtual elements created by the joins in this collection.
Data type: CDATA

Value: any string of characters.
Default: #IMPLIED
The desc attribute may be used to give a brief description of the virtual elements jointed in this <joinGrp> if their function and significance are not obvious.

Example:

```
<joinGrp result='q' targType='q' domains='zuitxt' >
  <join  targets='zuiq1 zuiq2 zuiq6'>
  <join  targets='zuiq3 zuiq4 zuiq5'>
</joinGrp>
```

Part: additional tag set for
Member of classes: pointerGroup [and indirectly also:] pointer
DTD file: teilink2.dtd
Data description: Any number of joins, pointers or extended pointers.
Occurs within: [none]
May contain: join ptr xptr
Declaration:

```
<!ELEMENT joinGrp       - -  ((join | ptr | xptr)*)             >

<!ATTLIST joinGrp            %a.global
                             %a.pointerGroup
          result             CDATA                #IMPLIED
          desc               CDATA                #IMPLIED       >
```

Discussed in: 14.7 ('Aggregation') on p. 369

keywords

<keywords> (i.e. Keywords) contains a list of keywords or phrases identifying the topic or nature of a text.

Attributes:

scheme identifies the controlled vocabulary within which the set of keywords concerned is defined.
Data type: IDREF
Value: identifier of the associated <taxonomy> element
Default: #IMPLIED

Example:

```
<keywords scheme=BL>
    <list><item>Babbage, Charles
  <item>Mathematicians - Great Britain - Biography
     </list>
</keywords>
```

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:
Occurs within: textClass
May contain: list term
Declaration:

```
<!ELEMENT keywords      - O  (term+ | list)                    >

<!ATTLIST keywords           %a.global
          scheme             IDREF                #IMPLIED       >
```

Discussed in: 5.4.3 ('The Text Classification') on p. 111

kinesic

<kinesic> (i.e. Non-vocalized quasi-lexical) any communicative phenomenon, not necessarily vocalized, for example a gesture, frown, etc.

Attributes:

who supplies an identifier for the participant performing the gesture. Its value is the identifier of a <participant> or <participant.grp> element in the TEI header.

Data type: IDREF
Value: Must identify a participant or participant group within the TEI Header
Default: %INHERITED

iterated (i.e. iterated) indicates whether or not the phenomenon is repeated.
Data type: (Y | N | U)
Sample values include:

y the phenomenon is repeated.
n the phenomenon is atomic.
u unknown or unmarked.

Default: N

desc (i.e. description) supplies a conventional representation for the phenomenon.
Data type: CDATA
Value: a description or representation of the phenomenon chosen from a semi-closed list
Default: #IMPLIED

Example:

```
<kinesic iterated=Y desc="shaking head vigorously" dur="1.5secs">
```

Part: base tag set for spoken materials
Member of classes: comp.spoken, timed
DTD file: teispok2
Data description: empty
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage u view
May contain: [none]
Declaration:

```
<!ELEMENT kinesic       - O  EMPTY                              >

<!ATTLIST kinesic            %a.global
                             %a.timed
          who                IDREF              %INHERITED

          iterated           (y | n | u)        n
          desc               CDATA              #IMPLIED        >
```

Discussed in: 11.2.7 ('Formal Definition') on p. 259; 11.2 ('Elements Unique to Spoken Texts') on p. 253; 11.2.3 ('Vocal, Kinesic, Event') on p. 256

l

<l> (i.e. verse line) contains a single, possibly incomplete, line of verse.

Attributes:

part specifies whether or not the line is metrically complete.
Data type: (Y | N | I | M | F)
Sample values include:

Y the line is metrically incomplete
N either the line is complete, or no claim is made as to its completeness
I the initial part of an incomplete line
M a medial part of an incomplete line
F the final part of an incomplete line

Default: N
The values I, M, or F should be used only where it is clear how the line is to be reconstituted.

Example:

```
<l part=Y met='-/-/-/-/-/'>
```

Part: base tag set for verse texts
Member of classes: chunk, enjamb, metrical [and indirectly also:] common
DTD file: teicore2
Data description: contains character data or phrase level elements only

Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item lg lg1 lg2 lg3 lg4 lg5 metDecl note performance prologue q quote remarks set sic sp stage view

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref

Declaration:

```
<!ELEMENT l             - O  (%paraContent)                  >
<!ATTLIST l                  %a.global
                             %a.enjamb
                             %a.metrical
          part               (Y | N | I | M | F) N            >
```

Discussed in: 6.11.1 ('Core Tags for Verse') on p. 177; 6.11 ('Passages of Verse or Drama') on p. 176; 10.2.4 ('Speech Contents') on p. 241

label

<label> contains the label associated with an item in a list; in glossaries, marks the term being defined.

Attributes: [None: global and inherited attributes only.]

Labels are most commonly used for the headwords in glossary lists; note the use of the global lang attribute to set the default language of the glossary list to Middle English, and identify the glosses and headings as modern English or Latin.

Example:

```
<list type=gloss lang=ME>
<head          lang=EN>Vocabulary</head>
<head.label lang=EN>Middle English</head.label>
<head.item  lang=EN>New English</head.item>
<label>nu         <item lang=EN>now
<label>lhude         <item lang=EN>loudly
<label>bloweth         <item lang=EN>blooms
<label>med         <item lang=EN>meadow
<label>wude         <item lang=EN>wood
<label>awe         <item lang=EN>ewe
<label>lhouth         <item lang=EN>lows
<label>sterteth  <item lang=EN>bounds, frisks
              (cf. <bibl>Chaucer, <title>K.T.</title>
              644</bibl>:  <q>a courser,
              <egwd>sterting</egwd> as the fyr</q>
<label>verteth         <item lang=LA>pedit
<label>murie         <item lang=EN>merrily
<label>swik         <item lang=EN>cease
<label>naver         <item lang=EN>never
</list>
```

Labels may also be used to record explicitly the numbers or letters which mark list items in ordered lists, as in this extract from Gibbon's *Autobiography*. In this usage the <label> element is synonymous with the n attribute on the <item> element.

Example:

```
I will add two facts, which have seldom occurred in
the composition of six, or at least of five quartos.
<list type=ordered rend=runon>
<label>(1) <item>My first rough manuscript, without any
intermediate copy, has been sent to the press.
<label>(2) <item>Not a sheet has been seen by any human
eyes, excepting those of the author and the printer:
the faults and the merits are exclusively my own.
</list>
```

Labels may also be used for other structured list items, as in this extract from the journal of Edward Gibbon:

Example:

```
<list type=gloss>
<label>March 1757. <item>I wrote some critical observations
upon Plautus.
<label>March 8th.  <item>I wrote a long dissertation upon
some lines of Virgil.
<label>June.           <item>I saw Mademoiselle Curchod --
<q lang=LA>Omnia vincit amor, et nos cedamus amori.</q>
<label>August.          <item>I went to Crassy, and staid two days.
</list>
```

Example:

```
<list type=gloss>
<label>The fourth volume of the <title>History of the
Decline and Fall of the Roman Empire</title>
<item>begun March 1, 1782 -- ended June, 1784.
<label>The fifth volume
<item>begun July, 1784 -- ended May 1, 1786.
<label>The sixth volume
<item>begun May 18, 1786 -- ended June 27, 1787.
</list>
```

Part: additional tag set for common core features
Member of classes: lists [and indirectly also:] common, inter
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: add admin argument body camera caption case castList cell colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem list meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT label          - O  (%phrase.seq;)                    >

<!ATTLIST label               %a.global                         >
```

Discussed in: 6.7 ('Lists') on p. 149

lacunaEnd

<lacunaEnd> indicates the end of a lacuna in a mostly complete textual witness.

Attributes: [None: global and inherited attributes only.]
Example:

```
<rdg wit=X><lacunaEnd>auctorite</rdg>
```

(optional)
Part: additional tag set for text criticism
Member of classes: fragmentary
DTD file: teitc2
Data description: Empty.
Occurs within: [none]
May contain: [none]
Declaration:

```
<!ELEMENT lacunaEnd      - O  EMPTY                             >

<!ATTLIST lacunaEnd           %a.global
                              %a.fragmentary                    >
```

Discussed in: 19.1.5 (‘Fragmentary Witnesses’) on p. 479

<lacunaStart> indicates the beginning of a lacuna in the text of a mostly complete textual witness. lacunaStart

Attributes: [None: global and inherited attributes only.]
(optional)
Part: additional tag set for text criticism
Member of classes: fragmentary
DTD file: teitc2
Data description: Empty.
Occurs within: [none]
May contain: [none]
Declaration:

```
<!ELEMENT lacunaStart   - O  EMPTY                          >
<!ATTLIST lacunaStart        %a.global
                             %a.fragmentary                 >
```

Discussed in: 19.1.5 (‘Fragmentary Witnesses’) on p. 479

<lang> (i.e. language name) name of a language mentioned in etymological or other linguistic discussion. lang

Attributes: [None: global and inherited attributes only.]
Example:

```
<entry>
    <form><orth>publish</orth> ... </form>
    <etym><lang>ME.</> <mentioned>publisshen</>,
        <lang>F.</> <mentioned>publier</>,
        <lang>L.</> <mentioned>publicare, publicatum</>.
        <xr>See <ref>public</>; cf. <ref>2d -ish</>.</xr>
    </etym>
    <!-- ... -->
</entry>
(From:  Webster's Second International)
```

Part: base tag set for printed dictionaries
Member of classes: data, dictionaries [and indirectly also:] phrase
DTD file: teidict2
Data description: May contain character data mixed with phrase-level elements.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT lang          - -  (%paraContent)                 >
<!ATTLIST lang               %a.global
                             %a.dictionaries                >
```

Discussed in: 12.3.4 ('Etymological Information') on p. 289

langKnown

<langKnown> (i.e. linguistic competence) contains an informal description of a person's competence in different languages, dialects, etc.

Attributes: [None: global and inherited attributes only.]
Example:

```
<langKnown>Spoken French
```

Part: auxiliary tag set for corpora and collections
Member of classes:
DTD file: teicorp2
Data description: May contain character data and phrase-level elements.
Occurs within: [none]
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT langKnown     - -  (%phrase.seq)                    >

<!ATTLIST langKnown           %a.global                       >
```

Discussed in: 23.2.2 ('The Participants Description') on p. 545

language

<language> identifies the language being described in the writing system declaration.

Attributes:

iso639 gives the standard language code from ISO 639.
Data type: CDATA
Value: any two- or three-letter code included included in ISO 639; if the language is not included in the list in ISO 639, the value should be given as the empty string.
Default: #REQUIRED

Example:

```
<language iso639=''>Various</language>
<language iso639=GRC>Classical Greek</language>
```

In general, the "language" associated with a given writing system declaration will be a natural language; it may however be a dialect, an artificial language, or some other semiotic system conveniently treated as a language.

The content of the <language> element is a description in prose of what language the WSD describes. Usually this will simply be the conventional name of the language; more information may however be included as needed.

Specialized writing system declarations which document a public character set or entity set suitable for encoding several languages, and which are intended for use as a base component within other language-specific writing system declarations, should identify the <language> to which they apply as "Various". Such writing systems should *not* be used directly by encoded texts; they should be named only in the <baseWSD> element of language-specific writing system declarations.

Part: auxiliary tag set for writing system declarations
Member of classes:
DTD file: teiwsd2
Data description: May contain character data only.
Occurs within: langUsage writingSystemDeclaration
May contain: #PCDATA
Declaration:

```
<!ELEMENT language      - O  (#PCDATA)                        >

<!ATTLIST language            %a.global
          iso639              CDATA                #REQUIRED  >
```

Discussed in: 25.2 ('Identifying the Language') on p. 573; 25.1 ('Overall Structure of Writing System Declaration') on p. 571

<language> characterizes a single language or sublanguage used within a text. language

Attributes:

id specifies the identifier for the *writing system declaration* for this language (e.g. *eng*, *fra*, *deu*)
Data type: ID
Value: should usually be a language identifier from ISO 639.
Default: #IMPLIED

wsd specifies the entity containing the *writing system declaration* used for representing texts in this language.
Data type: ENTITY
Value: must be the name of an entity containing a full writing system declaration
Default: #IMPLIED
The entity containing the writing system declaration will typically be an external file; it must be declared in the DTD.

usage specifies the approximate percentage (by volume) of the text which uses this language.
Data type: NUMBER
Value: a whole number between 1 and 100
Default: #IMPLIED

Example:

```
<language wsd=LA usage=100>Pig Latin
```

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description: Particularly for sublanguages, an informal prose characterization should be supplied as content for the element in addition to the specification implied by the WSD
Occurs within: langUsage writingSystemDeclaration
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT language      - O  (%phrase.seq)                    >

<!ATTLIST language           %a.analysis
                             %a.linking
                             %a.terminology
          n                  CDATA               #IMPLIED
          lang               IDREF               %INHERITED

          rend               CDATA               #IMPLIED
          id                 ID                  #IMPLIED
          wsd                ENTITY              #IMPLIED
          usage              NUMBER              #IMPLIED      >
```

Discussed in: 5.4.2 ('Language Usage') on p. 109

<langUsage> (i.e. language usage) describes the languages, sublanguages, registers, dialects etc. represented within a text. langUsage

Attributes: [None: global and inherited attributes only.]
Example:

```
<langUsage>
<language wsd=FR usage=60>Qu&eacute;becois
<language wsd=EN usage=20>Canadian business English
<language wsd=EN usage=20>British English
</langUsage>
```

Part: auxiliary tag set for TEI headers
Member of classes: declarable
DTD file: teihdr2
Data description: may contain either a simple prose description, or more formally one or more <language> elements
Occurs within: profileDesc
May contain: language p
Declaration:

```
<!ELEMENT langUsage      - O  (p | language)+                    >

<!ATTLIST langUsage           %a.global
                              %a.declarable                      >
```

Discussed in: 5.4.2 ('Language Usage') on p. 109; 5.4 ('The Profile Description') on p. 108; 23.3.2 ('Declarable Elements') on p. 552

lb

<lb> (i.e. line break) marks the start of a new (typographic) line in some edition or version of a text.

Attributes:

ed (i.e. edition) indicates the edition or version in which the line break is located at this point
Data type: CDATA
Value: Any string of characters; usually a siglum conventionally used for the edition.
Default: #IMPLIED
Example:

```
<lb ed=Riverside n=123>
```

n (i.e. number or name) indicates the number or other value associated with the line which follows the point of insertion of this <lb>.
Data type: CDATA
Value: Any string of characters.
Default: #IMPLIED
Encoders should adopt a clear and consistent policy as to whether the numbers associated with line breaks relate to the physical sequence number of the line within the page, or to some aspect of the logical structure of the text. By convention, <lb> elements should appear at the start of the line to which they refer.

Like other forms of milestone tag, <lb> tags cannot be automatically verified by SGML; for better validation, a concurrent markup stream should be used.

The <lb> tag is intended for making typographic line breaks in prose. It should be carefully distinguished from the <l> element, used to mark lines of verse.

Part: additional tag set for common core features
Member of classes: refsys
DTD file: teicore2
Occurs within: [none]
May contain: [none]
Declaration:

```
<!ELEMENT lb             - O  EMPTY                              >

<!ATTLIST lb                  %a.analysis
                              %a.linking
                              %a.terminology
          id                  ID                  #IMPLIED
          lang                IDREF               %INHERITED

          rend                CDATA               #IMPLIED
          ed                  CDATA               #IMPLIED
          n                   CDATA               #IMPLIED      >
```

Discussed in: 6.9.3 ('Milestone Tags') on p. 158; 10.2.4 ('Speech Contents') on p. 241

<lbl> (i.e. label) in dictionaries, contains a label for a form, example, translation, or other piece of information, e.g. “abbreviation for”, “contraction of”, “literally”, “approximately”, “synonyms:”, etc. lbl

Attributes:

type classifies the label using any convenient typology.
Data type: CDATA
Value: any string of characters, such as “usage”, “sense restriction”, etc.
Default: #IMPLIED

Labels specifically relating to usage should be tagged with the special-purpose <usg> element rather than with the generic <lbl> element.

Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, formInfo, gramInfo
DTD file: teidict2
Data description: May contain character data and phrase-level elements.
Occurs within: etym form gramGrp trans xr
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT lbl           - O  (%paraContent;)                  >

<!ATTLIST lbl                %a.global
                             %a.dictionaries
          type               CDATA               #IMPLIED    >
```

Discussed in: 12.3.1 (‘Information on Written and Spoken Forms’) on p. 279; 12.3.3.2 (‘Translation Equivalents’) on p. 287; 12.3.5.3 (‘Cross References to Other Entries’) on p. 293; 12.3.1 (‘Information on Written and Spoken Forms’) on p. 279

<leaf> encodes the leaves (terminal nodes) of a tree. leaf

Attributes:

label gives a label for a leaf.
Data type: CDATA
Value: A character string.
Default: #IMPLIED

value provides the value of a leaf, which is a feature structure or other analytic element.
Data type: IDREF
Value: A valid identifier of a feature structure or other analytic element.
Default: #IMPLIED

parent provides the ID of parent of a leaf.
Data type: IDREF
Value: The ID of the parent node.
Default: #IMPLIED

follow provides an ID of an element which this leaf follows.
Data type: IDREF
Value: An ID of another intermediate node or leaf of the tree.
Default: #IMPLIED
If the tree is unordered or partially ordered, this attribute has the property of fixing the relative order of the leaf and the element which is the value of the attribute.

Example:

```
<leaf id=peri1 parent=N1 label=periscope>
```

The in degree of a leaf is always 1, its out degree always 0.

Part: additional tag set for graphs, networks, and trees
Member of classes:
DTD file: teinet2

Data description: empty
Occurs within: tree
May contain: [none]
Declaration:

```
<!ELEMENT leaf          - O  EMPTY                          >
<!ATTLIST leaf               %a.global
          label              CDATA              #IMPLIED
          value              IDREF              #IMPLIED
          parent             IDREF              #IMPLIED
          follow             IDREF              #IMPLIED    >
```

Discussed in: 21.2 ('Trees') on p. 514

lem

<lem> (i.e. lemma) contains the lemma, or base text, of a textual variation.

Attributes: [None: global and inherited attributes only.]

The term *lemma* is used in text criticism to describe the reading in the text itself (as opposed to those in the apparatus); this usage is distinct from that of mathematics (where a lemma is a major step in a proof) and natural-language processing (where a lemma is the dictionary form associated with an inflected form in the running text).

Part: additional tag set for text criticism
Member of classes: readings
DTD file: teitc2
Data description: May contain character data and phrase-level elements.
Occurs within: app
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref [May include at any level: %m.fragmentary]
Declaration:

```
<!ELEMENT lem           - O  (%paraContent)      +(%m.fragmentary)
                                                                >
<!ATTLIST lem                %a.global
                             %a.readings                        >
```

Discussed in: 19.1 ('The Apparatus Entry, Readings, and Witnesses') on p. 469

lg

<lg> (i.e. line group) contains a group of verse lines functioning as a formal unit e.g. a stanza, refrain, verse paragraph, etc.

Attributes: [None: global and inherited attributes only.]
Example:

```
<lg type=free>
<l>Let me be my own fool
<l>of my own making, the sum of it
</lg>
<lg>
<l>is equivocal.
<l>One says of the drunken farmer:
</lg>
<lg>
<l>leave him lay off it. And this is
<l>the explanation.
</lg>
```

Part: base tag set for common core features
Member of classes: chunk, divn, metrical [and indirectly also:] common
DTD file: teicore2
Data description: contains verse lines or nested line groups only, possibly prefixed by a heading.

Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item lg metDecl note performance prologue q quote remarks set sic sp stage view
May contain: argument byline closer docAuthor docDate epigraph head l lg opener salute signed trailer
Declaration:

```
<!ELEMENT lg            - O  ((%m.divtop)*, (l | lg)+,
                              (%m.divbot)*)                    >
<!ATTLIST lg                 %a.global
                             %a.divn
                             %a.metrical                       >
```

Discussed in: 6.11.1 ('Core Tags for Verse') on p. 177; 6.11 ('Passages of Verse or Drama') on p. 176; 10.2.4 ('Speech Contents') on p. 241

<lg1> (i.e. line group) contains a first-level (i.e. largest) group of verse lines functioning as a formal unit e.g. a stanza, refrain, verse paragraph, etc. — lg1

Attributes: [None: global and inherited attributes only.]
Example:

```
<lg1 n='VIII' type=stanza>
   <lg2 type=sestet>
     <l>In the first year of Freedom's second dawn
     <l>Died George the Third; although no tyrant, one
     <l>Who shielded tyrants, till each sense withdrawn
     <l>Left him nor mental nor external sun:
     <l>A better farmer ne'er brushed dew from lawn,
     <l>A worse king never left a realm undone!
   </lg2>
   <lg2 type=couplet>
     <l>He died &dash; but left his subjects still behind,
     <l>One half as mad &dash; and t'other no less blind.
   </lg2>
</lg1>
```

Part: base tag set for verse texts
Member of classes: comp.verse, divn, metrical
DTD file: teivers2.dtd
Data description: contains verse lines or nested line groups only, possibly prefixed by a heading.
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage view
May contain: head l lg2
Declaration:

```
<!ELEMENT lg1           - O  (head?, (l | lg2)+)              >
<!ATTLIST lg1                %a.global
                             %a.divn
                             %a.metrical                       >
```

Discussed in: 9.2 ('Structural Divisions of Verse Texts') on p. 214

<lg2> (i.e. line group) contains a second-level (i.e. second largest) group of verse lines functioning as a formal unit e.g. a stanza, refrain, verse paragraph, etc. — lg2

Attributes: [None: global and inherited attributes only.]
Part: base tag set for verse texts
Member of classes: divn, metrical
DTD file: teivers2.dtd
Data description: contains verse lines or nested line groups only, possibly prefixed by a heading.
Occurs within: lg1
May contain: head l lg3
Declaration:

```
<!ELEMENT lg2          - O  (head?, (l | lg3)+)          >
<!ATTLIST lg2               %a.global
                            %a.divn
                            %a.metrical                  >
```

Discussed in: 9.2 ('Structural Divisions of Verse Texts') on p. 214

lg3

<lg3> (i.e. line group) contains a third-level (i.e. third largest) group of verse lines functioning as a formal unit e.g. a stanza, refrain, verse paragraph, etc.

Attributes: [None: global and inherited attributes only.]
Part: base tag set for verse texts
Member of classes: divn, metrical
DTD file: teivers2.dtd
Data description: contains verse lines or nested line groups only, possibly prefixed by a heading.
Occurs within: lg2
May contain: head l lg4
Declaration:

```
<!ELEMENT lg3          - O  (head?, (l | lg4)+)          >
<!ATTLIST lg3               %a.global
                            %a.divn
                            %a.metrical                  >
```

Discussed in: 9.2 ('Structural Divisions of Verse Texts') on p. 214

lg4

<lg4> (i.e. line group) contains a fourth-level (i.e. fourth largest) group of verse lines functioning as a formal unit e.g. a stanza, refrain, verse paragraph, etc.

Attributes: [None: global and inherited attributes only.]
Part: base tag set for verse texts
Member of classes: divn, metrical
DTD file: teivers2.dtd
Data description: contains verse lines or nested line groups only, possibly prefixed by a heading.
Occurs within: lg3
May contain: head l lg5
Declaration:

```
<!ELEMENT lg4          - O  (head?, (l | lg5)+)          >
<!ATTLIST lg4               %a.global
                            %a.divn
                            %a.metrical                  >
```

Discussed in: 9.2 ('Structural Divisions of Verse Texts') on p. 214

lg5

<lg5> (i.e. line group) contains a fifth-level (i.e. smallest) group of verse lines functioning as a formal unit e.g. a stanza, refrain, verse paragraph, etc.

Attributes: [None: global and inherited attributes only.]
Part: base tag set for verse texts
Member of classes: divn, metrical
DTD file: teivers2.dtd
Data description: contains verse lines only, possibly prefixed by a heading.
Occurs within: lg4
May contain: head l
Declaration:

```
<!ELEMENT lg5          - O  (head?, l+)                  >
<!ATTLIST lg5               %a.global
                            %a.divn
                            %a.metrical                  >
```

Discussed in: 9.2 ('Structural Divisions of Verse Texts') on p. 214

<line> contains one line of a reference edition. line

Attributes: [None: global and inherited attributes only.]
Example:

```
<(La)page n=32>
<(La)line n=1> ... [text of edition La, p. 32, l. 1]
<(La)line n=2> ... [text of edition La, p. 32, l. 2]
<(La)line n=3> ... [text of edition La, p. 32, l. 3]
<(La)line n=4> ... [text of edition La, p. 32, l. 4]
<(La)page n=33> ...
```

(optional)
Part: base tag set for
Member of classes:
DTD file: teipl2
Data description: May contain character data only.
Occurs within: col page
May contain: #PCDATA
Declaration:

```
<!ELEMENT line          - -  (#PCDATA)                    >
<!ATTLIST line                %a.global                   >
```

Discussed in: 31.6 ('Concurrent Markup for Pages and Lines') on p. 637

<link> defines an association or hypertextual link among elements or passages, of some type not more precisely specifiable by other elements. link

Attributes:

targets specifies the SGML identifiers of the elements or passages to be linked or associated.
Data type: IDREFS
Value: Each value specified must be the same as that specified as value for an ID attribute for some other element in the current SGML document.
Default: #REQUIRED

type categorizes the group of associations in some respect, using any convenient set of categories.
Data type: CDATA
Value: Where type is specified on a <linkGrp> element, it supplies the default value for the type attribute on each <link> included within the group.
Default: %INHERITED;

This element should only be used to encode associations not otherwise provided for by more specific elements.

The location of this element within a document has no significance, unless it is included within a <linkGrp>, in which case it may inherit the value of the type attribute from the value given on the <linkGrp>.

Part: additional tag set for
Member of classes: loc, pointer [and indirectly also:] phrase
DTD file: teilink2.dtd
Data description:
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cit cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem linkGrp locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set

settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: [none]

Declaration:

```
<!ELEMENT link          - O  EMPTY                              >

<!ATTLIST link               %a.global
          resp               CDATA                #IMPLIED
          crdate             CDATA                #IMPLIED
          targType           NAMES                #IMPLIED
          targOrder          (Y | N | U)          U
          evaluate           (all | one | none)   #IMPLIED
          targets            IDREFS               #REQUIRED
          type               CDATA                %INHERITED;    >
```

Discussed in: 14.1 ('Pointers') on p. 333

linkGrp

<linkGrp> defines a collection of associations or hypertextual links.

Attributes: [None: global and inherited attributes only.]

Example:

```
<linkGrp type='translation'>
<link target='s1 z1'>
<link target='s2 z2'>
 ...
<link target='s99 z99' type='paraphrase'>
</linkGrp>
```

A web or link group is an administrative convenience, which should be used to collect a set of links together for any purpose, not simply to supply a default value for the type attribute.

Part: additional tag set for

Member of classes: pointerGroup [and indirectly also:] pointer

DTD file: teilink2.dtd

Data description: May contain one or more <link> elements only, optionally with interspersed pointer elements.

Occurs within: [none]

May contain: link ptr xptr

Declaration:

```
<!ELEMENT linkGrp       - -  (link | ptr | xptr)+               >

<!ATTLIST linkGrp            %a.global
                             %a.pointerGroup                    >
```

Discussed in: 14.1 ('Pointers') on p. 333

list

<list> contains any sequence of items organized as a list.

Attributes:

type describes the form of the list.

Data type: CDATA

Sample values include:

ordered list items are numbered or lettered.

bulleted list items are marked with a "bullet" or other typographic device.

simple list items are not numbered or bulleted.

gloss each list item glosses some term or concept, which is given by a <label> element preceding the list item.

Default: SIMPLE

The formal syntax of the element declarations allows <label> tags to be omitted from lists tagged <list type=gloss>; this is however a semantic error.

Example:

```
<list type=ordered>
<item>a butcher
<item>a baker
<item>a candlestick maker, with
<list type=bullets>
<item>rings on his fingers
<item>bells on his toes
</list>
</list>
```

The following example treats the short numbered clauses of Anglo-Saxon legal codes as lists of items. The text is from an ordinance of King Athelstan (924-939).

Example:

```
<div1><head>Athelstan's Ordinance</head>
<list type=ordered>
<item n='1'>Concerning thieves.  First, that no thief is to be
spared who is caught with the stolen goods, [if he is] over
twelve years and [if the value of the goods is] over eightpence.
<list type=ordered>
<item n='1.1'>And if anyone does spare one, he is to pay for the
thief with his wergild -- and the thief is to be no nearer a
settlement on that account -- or to clear himself by an oath of
that amount.
<item n=1.2>If, however, he [the thief] wishes to defend himself
or to escape, he is not to be spared [whether younger or older
than twelve].
<item n=1.3>If a thief is put into prison, he is to be in prison
40 days, and he may then be redeemed with 120 shillings; and the
kindred are to stand surety for him that he will desist for ever.
<item n=1.4>And if he steals after that, they are to pay for him
with his wergild, or to bring him back there.
<item n=1.5>And if
he steals after that, they are to pay for him with his wergild,
whether to the king or to him to whom it rightly belongs; and
everyone of those who supported him is to pay 120 shillings to
the king as a fine.
</list>
<item n=2>Concerning lordless men.  And we pronounced about these
lordless men, from whom no justice can be obtained, that one
should order their kindred to fetch back such a person to justice
and to find him a lord in public meeting.
<list type=ordered>
<item n=2.1>And if they then will not, or cannot, produce him on
that appointed day, he is then to be a fugitive afterwards, and
he who encounters him is to strike him down as a thief.
<item n=2.2>And he who harbours him after that, is to pay for him
with his wergild or to clear himself by an oath of that amount.
</list>
<item n=3>Concerning the refusal of justice.  The lord who
refuses justice and upholds his guilty man, so that the king is
appealed to, is to repay the value of the goods and 120 shillings
to the king; and he who appeals to the king before he demands
justice as often as he ought, is to pay the same fine as the
other would have done, if he had refused him justice.
<list type=ordered>
<item n=3.1>And the lord who is an accessory to a theft by his
slave, and it becomes known about him, is to forfeit the slave
and be liable to his wergild on the first occasionp if he does it
more often, he is to be liable to pay all that he owns.
<item n=3.2>And likewise any of the king's treasurers or of our
reeves, who has been an accessory of thieves who have committed
```

```
theft, is to liable to the same.
</list>
<item n=4>Concerning treachery to a lord.  And we have pronounced
concerning treachery to a lord, that he [who is accused] is to
forfeit his life if he cannot deny it or is afterwards convicted
at the three-fold ordeal.
</list>
```

Note that nested lists have been used so the tagging mirrors the structure indicated by the two-level numbering of the clauses. The clauses could have been treated as a one-level list with irregular numbering, if desired.

Example:

```
<p>These decrees, most blessed Pope Hadrian, we propounded in the
public council ... and they confirmed them in our hand in your
stead with the sign of the Holy Cross, and afterwards inscribed
with a careful pen on the paper of this page, affixing thus the
sign of the Holy Cross.
<list type=simple>
<item>I, Eanbald, by the grace of God archbishop of the holy
church of York, have subscribed to the pious and catholic
validity of this document with the sign of the Holy Cross.
<item>I, &AElig;lfwold, king of the people across the Humber,
consenting have subscribed with the sign of the Holy Cross.
<item>I, Tilberht, prelate of the church of Hexham, rejoicing
have subscribed with the sign of the Holy Cross.
<item>I, Higbald, bishop of the church of Lindisfarne, obeying
have subscribed with the sign of the Holy Cross.
<item>I, Ethelbert, bishop of Candida Casa, suppliant, have
subscribed with thef sign of the Holy Cross.
<item>I, Ealdwulf, bishop of the church of Mayo, have subscribed
with devout will.
<item>I, &AElig;thelwine, bishop, have subscribed through
delegates.
<item>I, Sicga, 'patrician', have subscribed with serene mind
with the sign of the Holy Cross.
</list>
```

Part: additional tag set for common core features

Member of classes: lists [and indirectly also:] common, inter

DTD file: `teicore2`

Data description: May contain an optional heading followed by a series of items, or a series of label and item pairs, the latter being optionally preceded by one or two specialized headings.

Occurs within: add admin argument body camera caption case castList cell colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype keywords l lang lbl lem meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition revisionDesc seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref

May contain: head headItem headLabel item label

Declaration:

```
<!ELEMENT list          - -  (head?, ( (item+) | (headLabel?,
                             headItem?, (label, item)+)))       >

<!ATTLIST list               %a.global
          type               CDATA                simple        >
```

Discussed in: 6.7 ('Lists') on p. 149

listBibl

<listBibl> (i.e. citation list) contains a list of bibliographic citations of any kind.

Attributes: [None: global and inherited attributes only.]

Example:

```
<listBibl>
    <head>Works consulted</head>
    <bibl>Blain, Clements and Grundy: Feminist Companion to
          Literature in English (Yale, 1990)
    </bibl>
    <biblStruct>
    <analytic>
         <title>The Interesting story of the Children in the Wood</>
    <monogr>
         <title>The Penny Histories</title>
         <author>Victor E Neuberg</author>
         <imprint><publisher>OUP</><date>1968</>
    </biblStruct>
</listBibl>
```

Part: base tag set for common core features
Member of classes: declarable, lists [and indirectly also:] common, inter
DTD file: teicore2
Data description:
Occurs within: add admin argument body camera caption case castList cell colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound sourceDesc stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref
May contain: bibl biblFull biblStruct head trailer
Declaration:

```
<!ELEMENT listBibl      - -  (head?, (bibl | biblStruct |
                              biblFull)+, trailer?)              >

<!ATTLIST listBibl           %a.global
                             %a.declarable                      >
```

Discussed in: 6.10.1 ('Elements of Bibliographic References') on p. 163; 5.2.7 ('The Source Description') on p. 90; 23.3.2 ('Declarable Elements') on p. 552

<locale> (i.e. locale) contains a brief informal description of the nature of a place for example a room, a restaurant, a park bench etc. — locale

Attributes: [None: global and inherited attributes only.]
Example:

```
<locale>a fashionable restaurant
```

Part: auxiliary tag set for corpora and collections
Member of classes:
DTD file: teicorp2
Data description: May contain character data and phrase-level elements.
Occurs within: setting
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT locale        - O  (%phrase.seq)                      >

<!ATTLIST locale             %a.global                          >
```

Discussed in: 23.2.3 ('The Setting Description') on p. 549

<m> (i.e. morpheme) represents a grammatical morpheme. — m

Attributes:

baseform identifies the morpheme's base form.
Data type: CDATA

Value: a string of characters representing the spelling of the morpheme's base form.
Default: #IMPLIED

Example:

```
<w type=adjective><w type=noun><m type=prefix baseform=con>com</m>
<m type=root>fort</m></w><m type=suffix>able</m></w>
```

Part: additional tag set for
Member of classes: seg [and indirectly also:] phrase
DTD file: teiana2
Data description: May contain character data, <seg>, and <c> elements only.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view w wit witness witDetail writing xr xref
May contain: #PCDATA c seg
Declaration:

```
<!ELEMENT m              - -  ((#PCDATA | seg | c)*)              >

<!ATTLIST m                   %a.global
                              %a.seg
          baseform            CDATA                 #IMPLIED     >
```

Discussed in: 15.1 ('Linguistic Segment Categories') on p. 382

measure

<measure> contains a word or phrase referring to some quantity of an object or commodity, usually comprising a number, a unit and a commodity name.

Attributes:

type specifies the type of unit in which the measure is expressed.
Data type: CDATA
Sample values include:

weight measure of weight e.g. kg, pound.
count unit of count, e.g. dozen, score.
length measure of length, e.g. pole, mm.
area measure of area e.g. acre, hectare.
volume measure of volume e.g. litre, gallon.
currency unit of currency e.g. ecu, escudo, mark.

Default: #IMPLIED

Example:

```
<measure key=LBSF11 type='weight'>
    <num>2</num> pounds of flesh</measure>
<measure type=currency value='$1.56'>a quid</measure>
<measure type=area value='8 ECU'>2 merks of old extent</measure>
```

Part: additional tag set for common core features
Member of classes: data, names [and indirectly also:] phrase
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country

creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT measure       - -  (%phrase.seq;)                    >
<!ATTLIST measure            %a.global
                             %a.names
          type               CDATA               #IMPLIED     >
```

Discussed in: 6.4.3 ('Numbers and Measures') on p. 135

<meeting> in bibliographic references, contains a description of the meeting or conference from which the bibliographic item derives. meeting

Attributes: [None: global and inherited attributes only.]

Example:

```
<meeting>Ninth International Conference on Middle High German
Textual Criticism, Aachen, June 1998.</meeting>
```

Part: base tag set for common core features

Member of classes:

DTD file: teicore2

Occurs within: monogr

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref

Declaration:

```
<!ELEMENT meeting       - -  (%paraContent)                    >
<!ATTLIST meeting            %a.global                         >
```

Discussed in: 6.10.2.2 ('Authors, Titles, and Editors') on p. 168

<mentioned> marks words or phrases mentioned, not used. mentioned

Attributes: [None: global and inherited attributes only.]

Example:

```
There is thus a striking accentual difference between a verbal
form like <mentioned lang=EL id=c234>eluthemen</mentioned>
<gloss target=c234>we were released,</gloss> accented on
the second syllable of the word, and its
participial derivative <mentioned lang=EL id=c235>lutheis</mentioned>
<gloss target=c235>released,</gloss> accented on the last.
```

Part: base tag set for common core features

Member of classes: hqphrase [and indirectly also:] phrase

DTD file: teicore2

Data description: free prose

Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country

creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT mentioned     - -  (%phrase.seq;)                >
<!ATTLIST mentioned          %a.global                     >
```

Discussed in: 6.3.4 ('Terms, Glosses, and Cited Words') on p. 130

metDecl

<metDecl> documents the notation employed to represent a metrical pattern when this is specified as the value of a met, real, or rhyme attribute on any structural element of a metrical text (e.g. <lg>, <l>, or <seg>).

Attributes:

type indicates whether the notation conveys the abstract metrical form, its actual prosodic realization, or the rhyme scheme, or some combination thereof.
Data type: NAMES
Value: One or more of the three attribute names met, real, or rhyme.
Default: "MET REAL"
By default, the <metDecl> element documents the notation used for metrical pattern and realization. It may also be used to document the notation used for rhyme scheme information; if not otherwise documented, the rhyme scheme notation defaults to the traditional "abab" notation.

pattern specifies a regular expression defining any value that is legal for this notation.
Data type: CDATA
Value: the value must be a valid expression for the PATTERN keyword as defined in the TEI extended pointer notation (see section 14.2.2.14 ('The PATTERN Keyword') on p. 349).
Default: #IMPLIED
Example:

```
<metDecl pattern= '( (1|0)+  \|? /? )* ' >
<metDecl pattern= '((+-|-+)-+-+-+-+/)'
```

Example:

```
<metDecl id=ip pattern= '((+-|-+)-+-+-+-+/)'
     <symbol value='+'>stressed syllable
     <symbol value='-'>unstressed syllable
     <symbol value='/'>metrical line boundary
</metDecl>
```

This example is intended for the far more restricted case typified by the Shakespearean iambic pentameter. Only metrical patterns containing exactly ten syllables, alternately stressed and unstressed, (except for the first two which may be in either order) to each metrical line can be expressed using this notation.

The encoder may choose whether to define the notation formally or informally. However, the two methods may not be mixed.

Only usable within the header if the verse base is enabled.

Part: base tag set for TEI headers

Member of classes: declarable
DTD file: teihdr2
Data description: contains either a sequence of <symbol> elements or paragraphs. If one <symbol> is defined, then all the codes appearing within the pattern attribute should be documented, if the latter is specified.
Occurs within: encodingDesc
May contain: bibl biblFull biblStruct camera caption castList cit entry entryFree event eTree graph kinesic l label lg lg1 list listBibl move note p pause q quote shift sound sp stage superentry symbol tech termEntry tree u view vocal writing TEI...end
Declaration:

```
<!ELEMENT metDecl       - O  ((%component.seq) | (symbol+))     >

<!ATTLIST metDecl            %a.global
                             %a.declarable
          type               NAMES               "MET REAL"
          pattern            CDATA               #IMPLIED       >
```

Discussed in: 5.3.8 ('The Metrical Declaration Element') on p. 106; 9.4 ('Rhyme and Metrical Analysis') on p. 221

milestone

<milestone> marks the boundary between sections of a text, as indicated by changes in a standard reference system.

Attributes:

ed (i.e. edition) indicates which edition or version the milestone applies to.
Data type: CDATA
Value: Any string of characters; usually a siglum conventionally used for the edition.
Default: #IMPLIED

n (i.e. number or name) indicates the new number or other value for the unit which changes at this milestone.
Data type: CDATA
Value: Any string of characters.
Default: #IMPLIED
The special value *unnumbered* should be used in passages which fall outside the normal numbering scheme (e.g. chapter heads, poem numbers or titles, or speaker attributions in verse drama).

unit indicates what kind of section is changing at this milestone.
Data type: CDATA
Sample values include:

page page breaks in the reference edition.
column column breaks.
line line breaks.
book any units termed 'book', 'liber', etc.
poem individual poems in a collection.
canto cantos or other major sections of a poem.
stanza stanzas within a poem, book, or canto.
act acts within a play.
scene scenes within a play or act.
section sections of any kind.
absent passages not present in the reference edition.

Default: #REQUIRED
If the milestone marks the beginning of a piece of text not present in the reference edition, the special value *absent* may be used as the value of unit. The normal interpretation is that the reference edition does not contain the text which follows, until the next <milestone> tag for the edition in question is encountered.
In addition to the values suggested, other terms may be appropriate (e.g. *Stephanus* for the Stephanus numbers in Plato).

Example:

```
<milestone ed=La name=Dreissiger n=23>
...
<milestone ed=La name=Dreissiger n=24>
...
```

Milestones for page and column should precede milestones for line numbers, but this and other logical requirements cannot be enforced automatically by SGML; for better validation, a concurrent markup stream should be used.
Part: additional tag set for common core features
Member of classes: refsys
DTD file: teicore2
Occurs within: [none]
May contain: [none]
Declaration:

```
<!ELEMENT milestone     - O  EMPTY                              >

<!ATTLIST milestone          %a.analysis
                             %a.linking
                             %a.terminology
          id                 ID                  #IMPLIED
          lang               IDREF               %INHERITED

          rend               CDATA               #IMPLIED
          ed                 CDATA               #IMPLIED
          n                  CDATA               #IMPLIED
          unit               CDATA               #REQUIRED    >
```

Discussed in: 6.9.3 ('Milestone Tags') on p. 158

minus

<minus> (i.e. Binary minus value) provides binary minus value for a feature.

Attributes: [None: global and inherited attributes only.]
Example:

```
<f name=nasal><minus></f>
```

Part: additional tag set for feature structures
Member of classes: binary [and indirectly also:] singleVal
DTD file: teifs2
Data description: Empty element.
Occurs within: f fvLib vAlt
May contain: [none]
Declaration:

```
<!ELEMENT minus         - O  EMPTY                              >

<!ATTLIST minus              %a.global                          >
```

Discussed in: 16.2 ('Elementary Feature Structures: Features with Binary Values') on p. 398

minute

<minute> (i.e. minute) the minute component of a structured time-expression.

Attributes: [None: global and inherited attributes only.]
Example:

```
The train leaves for Boston at
<timeStruct value='1345'>
<hour>13</hour>:
<minute>45</minute>
</timeStruct>
```

Part: additional tag set for names and dates
Member of classes: temporalExpr
DTD file: teind2
Occurs within: dateStruct timeStruct

May contain: #PCDATA
Declaration:

```
<!ELEMENT minute        - -  (#PCDATA)                          >

<!ATTLIST minute             %a.global
                             %a.temporalExpr                    >
```

Discussed in: 20.4 ('Dates and Time') on p. 499

<monogr> (i.e. monographic level) contains bibliographic elements describing an item (e.g. a book or journal) published as an independent item (i.e. as a separate physical object). monogr

Attributes: [None: global and inherited attributes only.]
The <monogr> element may occur only within bibliographic citation or reference elements; it is mandatory for description of the monographic level of <bibl.struct> elements.
Part: base tag set for common core features
Member of classes: biblPart
DTD file: teicore2
Data description: May contain specialized bibliographic elements, in a prescribed order.
Occurs within: bibl biblStruct
May contain: author biblScope edition editor extent imprint meeting note respStmt title
Declaration:

```
<!ELEMENT monogr        - O  ( ( ((author | editor |
                             respStmt)+, title+, (editor |
                             respStmt)*) | (title+, (author |
                             editor | respStmt)*))?, (note |
                             meeting)*, (edition, (editor |
                             respStmt)*)*, imprint, (imprint |
                             extent | biblScope)* )          >

<!ATTLIST monogr             %a.global                       >
```

Discussed in: 6.10.2.1 ('Analytic, Monographic, and Series Levels') on p. 166

<month> (i.e. month) the month component of a structured date. month

Attributes: [None: global and inherited attributes only.]
Example:

```
<dateStruct value='14-05-1993'>
<day type=name>Friday</day>,
<day type=number>14</day>
<month type=name>May</month>
<year>1993</year></dateStruct>
```

Part: additional tag set for names and dates
Member of classes: temporalExpr
DTD file: teind2
Occurs within: dateStruct timeStruct
May contain: #PCDATA
Declaration:

```
<!ELEMENT month         - -  (#PCDATA)                          >

<!ATTLIST month              %a.global
                             %a.temporalExpr                    >
```

Discussed in: 20.4 ('Dates and Time') on p. 499

<mood> contains information about the grammatical mood of verbs (e.g. "indicative", "subjunctive", "imperative") mood

Attributes: [None: global and inherited attributes only.]
This element is synonymous with <gram type=mood>.
Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, morphInfo

DTD file: teidict2
Data description: May contain character data and phrase-level elements.
Occurs within: etym trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT mood          - O  (%paraContent;)                >

<!ATTLIST mood              %a.global
                            %a.dictionaries                 >
```

Discussed in: 12.3.1 ('Information on Written and Spoken Forms') on p. 279

move

<move> (i.e. Movement) marks the actual entrance or exit of one or more characters on stage.
Attributes:

who identifies the character or characters performing the movement.
Data type: IDREFS
Value: The IDREFS are derived from the ID attribute on the role elements in the cast list.
Default: #REQUIRED

type characterizes the movement, for example as an entrance or exit.
Data type: CDATA
Sample values include:

entrance character is entering the stage.
exit character is exiting the stage.
onstage character moves on stage

Default: #IMPLIED

where specifies the direction of a stage movement.
Data type: CDATA
Sample values include:

L stage left
R stage right
C centre stage

Default: #IMPLIED
Full blocking information will normally require combinations of values, (for example "UL" for "upper stage left") and may also require more detailed encoding of speed, direction etc. Full documentation of any coding system used should be provided in the header.

perf identifies the performance or performances in which this movement occurred as specified.
Data type: IDREFS
Value: The ID of a <performance> element.
Default: #IMPLIED

Example:

```
<stage type="entrance">
<move who=B type="enter" where=SL perf=P1>
Enter Bellafront mad.</stage>
```

Part: base tag set for performance texts
Member of classes: stageDirection [and indirectly also:] comp.drama, inter
DTD file: teidram2
Data description: Empty
Occurs within: add admin argument body camera caption case castList cell colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood

note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref

May contain: [none]

Declaration:

```
<!ELEMENT move          - O  EMPTY                              >

<!ATTLIST move                %a.global
          who                 IDREFS              #REQUIRED
          type                CDATA               #IMPLIED
          where               CDATA               #IMPLIED
          perf                IDREFS              #IMPLIED      >
```

Discussed in: 10.2.3 ('Stage Directions') on p. 238

<msr> (i.e. Measure value) provides a measure value or range of values for a feature. msr

Attributes:

value provides a numeric value.
Data type: CDATA
Value: A real number or integer.
Default: #REQUIRED

valueTo together with value attribute, provides a range of numeric values.
Data type: CDATA
Value: A real number or integer.
Default: #IMPLIED

unit provides a unit for a measure feature, one of a finite list that may be specified in a feature declaration.
Data type: CDATA
Value: A string, e.g. *meter*.
Default: #REQUIRED

rel indicates the relation of the given value or range to the actual value or range.
Data type: (EQ | NE | LT | LE | GT | GE)
Sample values include:

eq indicates that the actual value or range is that given.
ne indicates that the actual value or range is not the value or range given.
lt indicates that the actual value or range is less than the given value or range.
le indicates that the actual value or range is less than or equal to the given value or range.
gt indicates that the actual value or range is greater than the given value or range.
ge indicates that the actual value or range is greater than or equal to the given value or range.

Default: EQ

type indicates whether value or range is to be understood as real or integer.
Data type: (INT | REAL)
Sample values include:

int specifies that value is an integer; if noninteger is given as value of value, then only integer part is used.
real specifies that value is a real number.

Default: #IMPLIED

Example:

```
<msr value=10000 valueTo=20000 unit=guilder>
```

Part: additional tag set for feature structures
Member of classes: singleVal [and indirectly also:] featureVal
DTD file: teifs2
Data description: Empty tag.

Occurs within: f fvLib if vAlt vDefault vRange
May contain: [none]
Declaration:

```
<!ELEMENT msr           - O  EMPTY                              >

<!ATTLIST msr                %a.global
          value              CDATA                 #REQUIRED
          valueTo            CDATA                 #IMPLIED
          unit               CDATA                 #REQUIRED
          rel                (eq | ne | lt | le | gt | ge)
                                                   eq
          type               (int | real)          #IMPLIED     >
```

Discussed in: 16.4 ('Symbolic, Numeric, Measurement, Rate and String Values') on p. 402

name

<name> (i.e. name, proper noun) contains a proper noun or noun phrase.

Attributes:

type indicates the type of the object which is being named by the phrase.
Data type: CDATA
Value: Values such as person, place, institution, product, acronym.
Default: #IMPLIED

Proper nouns referring to people, places, and organizations may be tagged instead with <person>, <place>, or <org>.

Part: additional tag set for common core features
Member of classes: agent, data, names [and indirectly also:] phrase
DTD file: dummy teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateline dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName geogName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp respStmt restore role roleDesc roleName rs s salute seg sense set setting settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT name          - -  (%phrase.seq;)                     >

<!ATTLIST name               %a.global
                             %a.names
          type               CDATA                 #IMPLIED     >
```

Discussed in: 6.4.1 ('Referring Strings') on p. 132

nameLink

<nameLink> (i.e. name link) contains a connecting phrase or link used within a name but not regarded as part of it, such as "van der" or "of".

Attributes: [None: global and inherited attributes only.]
Example:

```
<persName>
  <foreName>Frederick</foreName>
  <nameLink>van der</nameLink>
```

```
  <surname>Tronck</surname>
</persName>
```

Example:

```
<persName>
  <foreName>Alfred</foreName>
  <nameLink>de</nameLink>
  <surname>Musset</surname>
</persName>
```

Part: additional tag set for names and dates
Member of classes: personPart [and indirectly also:] names
DTD file: teind2
Occurs within: persName
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT nameLink      - -  (%phrase.seq;)                >
<!ATTLIST nameLink           %a.global
                             %a.personPart                 >
```

Discussed in: 20.1 ('Personal Names') on p. 488

<nbr> (i.e. Numeric value) provides a numeric value or range of values for a feature. nbr

Attributes:

value provides a numeric value.
Data type: CDATA
Value: A real number or integer.
Default: #REQUIRED

valueTo together with value attribute, provides a range of numeric values.
Data type: CDATA
Value: A real number or integer.
Default: #IMPLIED

rel indicates the relation of the given value or range to the actual value or range.
Data type: (EQ | NE | LT | LE | GT | GE)
Sample values include:

- *eq* indicates that the actual value or range is that given.
- *ne* indicates that the actual value or range is not the value or range given.
- *lt* indicates that the actual value or range is less than the given value or range.
- *le* indicates that the actual value or range is less than or equal to the given value or range.
- *gt* indicates that the actual value or range is greater than the given value or range.
- *ge* indicates that the actual value or range is greater than or equal to the given value or range.

Default: EQ

type indicates whether value or range is to be understood as real or integer.
Data type: (INT | REAL)
Sample values include:

- *int* specifies that value is an integer; if noninteger is given as value of value, then only integer part is used.
- *real* specifies that value is a real number.

Default: #IMPLIED

Example:

```
<nbr type=int rel=ge value=0>
```

Part: additional tag set for feature structures
Member of classes: singleVal [and indirectly also:] featureVal
DTD file: teifs2
Data description: Empty tag.
Occurs within: f fvLib if vAlt vDefault vRange
May contain: [none]
Declaration:

```
<!ELEMENT nbr           - O  EMPTY                          >

<!ATTLIST nbr                %a.global
          value              CDATA               #REQUIRED
          valueTo            CDATA               #IMPLIED
          rel                (eq | ne | lt | le | gt | ge)
                                                 eq
          type               (int | real)        #IMPLIED       >
```

Discussed in: 16.4 ('Symbolic, Numeric, Measurement, Rate and String Values') on p. 402

node

<node> encodes a node, a possibly labeled point in a graph.

Attributes:

label gives a label for a node.
Data type: CDATA
Value: A character string.
Default: #IMPLIED

label2 gives a second label for a node.
Data type: CDATA
Value: A character string.
Default: #IMPLIED
Use this attribute together with the label attribute if a transducer is being encoded whose actions are associated with nodes rather than with arcs.

value provides the value of a node, which is a feature structure or other analytic element.
Data type: IDREF
Value: A valid identifier.
Default: #IMPLIED

type provides a type for a node.
Data type: CDATA
Sample values include:

initial initial node in a transition network
final final node in a transition network

Default: #IMPLIED

adjTo gives the IDs of the nodes which are adjacent to the current node.
Data type: IDREFS
Value: A list of IDs.
Default: #IMPLIED

adjFrom gives the IDs of the nodes which are adjacent from the current node.
Data type: IDREFS
Value: A list of IDs.
Default: #IMPLIED

adj gives the IDs of the nodes which are both adjacent to and adjacent from the current node.
Data type: IDREFS
Value: A list of IDs.
Default: #IMPLIED
Use this attribute instead of the adjTo and adjFrom attributes when the graph is undirected and vice versa if the graph is directed.

inDegree gives the in degree of the node, the number of nodes which are adjacent from the given node.

Data type: NUMBER
Value: A nonnegative integer.
Default: #IMPLIED

outDegree gives the out degree of the node, the number of nodes which are adjacent to the given node.
Data type: NUMBER
Value: A nonnegative integer.
Default: #IMPLIED

degree gives the degree of the node, the number of arcs with which the node is incident.
Data type: NUMBER
Value: A nonnegative integer.
Default: #IMPLIED
Use this attribute instead of the inDegree and outDegree attributes when the graph is undirected and vice versa if the graph is directed.

Example:

```
<node id=t6 label='6' inDegree=2 outDegree=0 type='final'>
```

Part: additional tag set for graphs, networks, and trees
Member of classes:
DTD file: teinet2
Data description: empty
Occurs within: graph
May contain: [none]
Declaration:

```
<!ELEMENT node          - O  EMPTY                          >
<!ATTLIST node               %a.global
          label              CDATA              #IMPLIED
          label2             CDATA              #IMPLIED
          value              IDREF              #IMPLIED
          type               CDATA              #IMPLIED
          adjTo              IDREFS             #IMPLIED
          adjFrom            IDREFS             #IMPLIED
          adj                IDREFS             #IMPLIED
          inDegree           NUMBER             #IMPLIED
          outDegree          NUMBER             #IMPLIED
          degree             NUMBER             #IMPLIED      >
```

Discussed in: 21.1 ('Graphs and Digraphs') on p. 506

<none> (i.e. None value) represents boolean *false* value variable. — none

Attributes: [None: global and inherited attributes only.]
Example:

```
<f name=inflection><none>
```

Part: additional tag set for feature structures
Member of classes: boolean [and indirectly also:] singleVal
DTD file: teifs2
Data description: Empty element.
Occurs within: f fvLib vAlt
May contain: [none]
Declaration:

```
<!ELEMENT none          - O  EMPTY                          >
<!ATTLIST none               %a.global                      >
```

Discussed in: 16.8 ('Boolean, Default and Uncertain Values') on p. 417

<normalization> indicates the extent of normalization or regularization of the original source carried out in converting it to electronic form. — normalization

Attributes:

source indicates the authority for any normalization carried out.
Data type: CDATA
Value: Should really be a bibliographic reference of some kind
Default: #IMPLIED

method indicates the method adopted to indicate normalizations within the text.
Data type: (SILENT | TAGS)
Sample values include:

silent normalization made silently

tags normalization represented using editorial tags

Default: SILENT

Example:

```
<editorialDecl>
  <normalization id=WP5 source="WP5.1 spelling dict">
    <p>Converted to modern American spelling </normalization>
  <normalization id=NOED source=OED method=tags>
</editorialDecl>
```

Part: auxiliary tag set for TEI headers
Member of classes: declarable
DTD file: teihdr2
Data description:
Occurs within: editorialDecl
May contain: p
Declaration:

```
<!ELEMENT normalization - O  (p+)                              >

<!ATTLIST normalization      %a.global
                             %a.declarable
           source            CDATA              #IMPLIED
           method            (silent | tags)    silent         >
```

Discussed in: 5.3.3 ('The Editorial Practices Declaration') on p. 96; 23.3.2 ('Declarable Elements') on p. 552

note

<note> (in a writing system) contains a note of any type.

Attributes: [None: global and inherited attributes only.]
Example:

```
<note>This writing system declaration describes the ad hoc
 character set developed for encoding Hebrew and Arabic for the
 Geniza Project at Princeton University.  It is intended for
 documentation of existing files only and should not be used to
 guide the implementation of new systems.
 </note>
```

A note may also relate to a single character:

Example:

```
<note>This character is the Japanese form of the unified Han
 character xxxx in ISO 10646; for the Korean form, use the
 preceding character.</note>
```

Part: auxiliary tag set for writing system declarations
Member of classes:
DTD file: teiwsd2
Data description: May contain character data only.
Occurs within: biblStruct character form formula monogr notesStmt writingSystemDeclaration
May contain: #PCDATA
Declaration:

```
<!ELEMENT note          - O  (#PCDATA)                         >
```

```
<!ATTLIST note                %a.global                        >
```

Discussed in: 25.5 ('Notes in the WSD') on p. 581; 25.1 ('Overall Structure of Writing System Declaration') on p. 571

<note> contains a note or annotation. note

Attributes:

n (i.e. number or symbol) indicates the symbol or number used to mark the note's point of attachment to the main text.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED
Example:

```
Mevorakh b. Saadya's mother, the matriarch of the family
during the second half of the eleventh century,
   <note n=126>
        The alleged mention of Judah Nagid's mother
        in a letter from 1071 is, in fact, a
        reference to Judah's children; cf. above,
        nn. 111 and 54.
   </note>
is well known from Geniza documents published by Jacob Mann.
```

If notes are numbered in sequence and their numbering can be reconstructed automatically by processing software, it may not be considered necessary to record the note numbers.

type describes the type of note.
Data type: CDATA
Value: Values can be taken from any convenient typology of annotation suitable to the work in hand; e.g. annotation, gloss, citation, digression, preliminary, temporary, ...
Default: #IMPLIED

resp (i.e. responsible) indicates who is responsible for the annotation: author, editor, translator, etc.
Data type: CDATA
Sample values include:

auth[or] note originated with the author of the text.
ed[itor] note added by the editor of the text.
comp[iler] note added by the compiler of a collection.
tr[anslator] note added by the translator of a text.
transcr[iber] note added by the transcriber of a text into electronic form.
(initials) note added by the individual indicated by the initials.

Default: #IMPLIED
For specialized types of editorial annotation (e.g. for marking corrections, normalizations, cruxes, etc.), see chapter 19 ('Critical Apparatus') on p. 467.

place indicates where the note appears in the source text.
Data type: CDATA
Sample values include:

foot note appears at foot of page.
end note appears at end of chapter or volume.
inline note appears as a marked paragraph in the body of the text.
left note appears in left margin.
right note appears in right margin.
interlinear note appears between lines of the text.
app[aratus] note appears in the apparatus at the foot of the page.

Default: 'UNSPECIFIED'
For pages with multiple apparatus, values such as *app1* and *app2* can be used.

The place attribute can be used to indicate to text formatting software where a note should be printed. If the locations indicated do not agree with those in the copy text, that fact should be indicated in the TEI header.

anchored indicates whether the copy text shows the exact place of reference for the note.
Data type: (YES | NO)
Sample values include:

yes copy text indicates the place of attachment for the note.

no copy text indicates no place of attachment for the note.

Default: YES

In modern texts, notes are usually anchored by means of explicit footnote or endnote symbols. An explicit indication of the phrase or line annotated may however be used instead (e.g. "page 218, lines 3-4"). The anchored attribute indicates whether any explicit location is given, whether by symbol or by prose cross-reference. If the specific symbols used are to be recorded, use the n attribute.

target indicates the point of attachment of a note, or the beginning of the span to which the note is attached.
Data type: IDREFS
Value: reference to the ids of element(s) which begin at the location in question (e.g. the id of an <anchor> element).
Default: #IMPLIED
If target and targetEnd are to be used to indicate where notes attach to the text, then elements at the appropriate locations (<anchor> elements if necessary) must be given id values to be pointed at.

targetEnd points to the end of the span to which the note is attached, if the note is not embedded in the text at that point.
Data type: IDREFS
Value: reference to the id(s) of element(s) which *end* at the location(s) in question, or to an empty element at the point in question.
Default: #IMPLIED

Example:

```
And yet it is not only in the great line of Italian
renaissance art, but even in the painterly
    <note resp=Tr><term lang=DE>Malerisch</term>.  This
    word has, in the German, two distinct meanings, one
    objective, a quality residing in the object, the
    other subjective, a mode of apprehension and creation.
    To avoid confusion, they have been distinguished in
    English as <mentioned>picturesque</> and
    <mentioned>painterly</> respectively.  (Tr.)</note>
style of the Dutch genre painters of the seventeenth
century that drapery has this psychological significance.
```

Part: additional tag set for common core features
Member of classes: biblPart, dictionaryTopLevel, notes, terminologyInclusions [and indirectly also:] common, inter
DTD file: teicore2 dummy
Data description: May contain character data and phrase-level elements.
Occurs within: add admin argument bibl biblStruct body camera caption case castList cell character colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph entry epigraph epilogue equiv etym figDesc foreign form formula fsDescr fDescr gen gram gramGrp head hi hom hyph imprimatur item itype l lang lbl lem meeting metDecl monogr mood note notesStmt number orth otherForm p per performance pos prologue pron q quote rdg re ref region remarks rendition seg sense set sic sound stage stress subc supplied syll tagUsage tech termEntry title titlePart tns tr trans unclear usg view witness witDetail writing writingSystemDeclaration xr xref
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph entry entryFree event expan eTree figure foreign formula gap gi gloss graph handShift hi kinesic l label lang lg lg1 link list listBibl m measure mentioned move name note num orig

oRef oVar p pause phr ptr pRef pVar q quote ref reg rs s seg shift sic sound soCalled sp stage superentry table tag tech term termEntry text time timeRange timeStruct title tree u val view vocal w writing xptr xref TEI...end

Declaration:

```
<!ELEMENT note          - O  (%specialPara;)                    >

<!ATTLIST note               %a.analysis
                             %a.linking
                             %a.terminology
          id                 ID                  #IMPLIED
          lang               IDREF               %INHERITED

          rend               CDATA               #IMPLIED
          n                  CDATA               #IMPLIED
          type               CDATA               #IMPLIED
          resp               CDATA               #IMPLIED
          place              CDATA               'unspecified'
          anchored           (yes | no)          yes
          target             IDREFS              #IMPLIED
          targetEnd          IDREFS              #IMPLIED       >
```

Discussed in: 6.8.1 ('Notes and Simple Annotation') on p. 152; 5.2.6 ('The Notes Statement') on p. 88; 6.10.2.5 ('Notes and Other Additional Information') on p. 174; 12.3.5.4 ('Notes within Entries') on p. 295; 13.2 ('Tags for Terminological Data') on p. 312

notesStmt

<notesStmt> (i.e. Notes statement) collects together any notes providing information about a text additional to that recorded in other parts of the bibliographic description.

Attributes: [None: global and inherited attributes only.]

Example:

```
<notesStmt>
  <note>Historical commentary provided by Mark Cohen</note>
  <note>OCR scanning done at University of Toronto</note>
</notesStmt>
```

Information of different kinds should not be grouped together into the same note.

Part: auxiliary tag set for TEI headers

Member of classes:

DTD file: teihdr2

Data description:

Occurs within: biblFull fileDesc

May contain: note

Declaration:

```
<!ELEMENT notesStmt     - O  (note+)                            >

<!ATTLIST notesStmt          %a.global                          >
```

Discussed in: 5.2.6 ('The Notes Statement') on p. 88; 5.2 ('The File Description') on p. 80

null

<null> (i.e. Null value) represents the null set if org=set is specified for the feature of which it is the value; represents the null bag if org=bag is specified for the feature of which it is the value; represents the null list if org=list is specified for the feature of which it is the value; has no interpretation if org=single is specified for the feature of which it is the value.

Attributes: [None: global and inherited attributes only.]

Example:

```
<f name=empty org=list><null>
```

If the <null> element occurs as the value of a feature, it must be the only value. It is also a semantic error for the org attribute of that feature to have the value *single*.

Part: additional tag set for feature structures

Member of classes: featureVal

DTD file: teifs2

Data description: Empty element.

Occurs within: f fvLib if vAlt vDefault vRange
May contain: [none]
Declaration:

```
<!ELEMENT null          - O  EMPTY                          >
<!ATTLIST null                %a.global                     >
```

Discussed in: 16.6 ('Singleton, Set, Bag and List Collections of Values') on p. 410

num

<num> (i.e. number) contains a number, written in any form.

Attributes:

type indicates the type of numeric value.
Data type: CDATA
Sample values include:
cardinal absolute number, e.g. "21", "21.5"
ordinal ordinal number, e.g. "21st"
fraction fraction, e.g. "one half" or "three halves"
percentage e.g. "ten percent"
Default: #IMPLIED
If a different typology is desired, other values can be used for this attribute.

value supplies the value of the number in an application-dependent standard form.
Data type: CDATA
Value: any numeric value in the chosen standard form.
Default: #IMPLIED
The standard form used should be described in the <std.vals> element in the TEI header. Standard forms may be defined from scratch, or borrowed from existing practice (e.g. "Standard values are given in the notation defined for numeric constants in the C language.")

Example:

```
<num type=cardinal value='21'>twenty-one</num>
<num type=cardinal value='1.5'>1.5</num>
He stands <num type=cardinal value='1.9'>1 &middot; 90</num>m. high.
```

Detailed analyses of quantities and units of measure in historical documents should use the feature structure mechanism described in chapter 16 ('Feature Structures') on p. 397. The <num> element is intended for use in simple applications.
Part: additional tag set for common core features
Member of classes: data [and indirectly also:] phrase
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT num           - -  (%phrase.seq;)                 >
```

```
<!ATTLIST num              %a.global
          type             CDATA              #IMPLIED
          value            CDATA              #IMPLIED      >
```

Discussed in: 6.4.3 ('Numbers and Measures') on p. 135

<number> indicates grammatical number associated with a form, as given in a dictionary. number

Attributes: [None: global and inherited attributes only.]

This element is synonymous with <gram type=num>.

Part: base tag set for printed dictionaries

Member of classes: dictionaries, dictionaryParts, morphInfo

DTD file: teidict2

Data description: May contain character data and phrase-level elements.

Occurs within: etym trans

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref

Declaration:

```
<!ELEMENT number        - -  (%paraContent;)              >

<!ATTLIST number             %a.global
                             %a.dictionaries              >
```

Discussed in: 12.3.1 ('Information on Written and Spoken Forms') on p. 279; 12.3.2 ('Grammatical Information') on p. 284

<occasion> a temporal expression (either a date or a time) given in terms of a named occasion such as a holiday, a named time of day, or some notable event. occasion

Attributes: [None: global and inherited attributes only.]

Example:

```
In New York,
  <dateStruct value='1-1'>
    <occasion type=holiday>New Years Day</occasion>
  </dateStruct> is the quietest of holidays,
  <dateStruct value='4 July'>
    <occasion type=holiday>Independence Day</occasion>
  </dateStruct>
the most turbulent.
```

Part: additional tag set for names and dates

Member of classes: temporalExpr

DTD file: teind2

Occurs within: dateStruct timeStruct

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT occasion      - -  (%phrase.seq)                >

<!ATTLIST occasion           %a.global
                             %a.temporalExpr              >
```

Discussed in: 20.4 ('Dates and Time') on p. 499

<occupation> contains an informal description of a person's trade, profession or occupation. occupation

Attributes:

scheme identifies the classification system or taxonomy in use by supplying the identifier of a <taxonomy> element elsewhere in the header.
Data type: IDREF

Value: must identify a <taxonomy> element
Default: #IMPLIED

code identifies an occupation code defined within the classification system or taxonomy defined by the source attribute.
Data type: IDREF
Value: Must identify a <category> element
Default: #IMPLIED

Example:

```
<occupation>accountant</occupation>
```

Example:

```
<occupation source=RG code=ACC>
```

Example:

```
<occupation source=RG code=ACC>
      accountant with specialist knowledge of oil industry
</occupation>
```

The content of this element may be used as an alternative to the more formal specification made possible by its attributes; it may also be used to supplement the formal specification with commentary or clarification.
Part: auxiliary tag set for corpora and collections
Member of classes: demographic
DTD file: teicorp2
Data description: May contain character data and phrase-level elements.
Occurs within: person personGrp
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT occupation    - -  (%phrase.seq)                    >

<!ATTLIST occupation         %a.global
          scheme             IDREF              #IMPLIED
          code               IDREF              #IMPLIED      >
```

Discussed in: 23.2.2 ('The Participants Description') on p. 545

offset

<offset> (i.e. offset) that part of a relative temporal or spatial expression which indicates the direction of the offset between the two place names, dates, or times involved in the expression.

Attributes: [None: global and inherited attributes only.]
Part: auxiliary tag set for names and dates
Member of classes: placePart, temporalExpr [and indirectly also:] names
DTD file: teind2
Occurs within: dateStruct placeName timeStruct
May contain: #PCDATA
Declaration:

```
<!ELEMENT offset        - -  (#PCDATA)                        >

<!ATTLIST offset             %a.global
          value              CDATA              #IMPLIED
                             %a.placePart                     >
```

Discussed in: 20.2 ('Place Names') on p. 493

ofig

<ofig> (i.e. other-form information group) within a <tig> element, contains information elements relating to a single <otherForm>.

Attributes:

type classifies the other-form information group according to some convenient typology, preferably the dictionary of data element types specified in ISO WD 12 620.

Data type: CDATA
Value: any string identifying a class of <ofig>
Default: #IMPLIED
A much fuller list of values for the type attribute may be generated from the dictionary of data element types under preparation as ISO TC 37/SC 3/WD 12 620, Computational Aids in Terminology. See ISO 12 620 for fuller details.

Part: base tag set for terminological data
Member of classes:
DTD file: teite2n
Data description: May contain an <otherForm> element and any data elements relating to the <otherForm> or to one of its related elements.
Occurs within: tig
May contain: admin descrip gram otherForm
Declaration:

```
<!ELEMENT ofig          - O  ((%m.terminologyMisc)*,
                              (otherForm, gram*),
                              (%m.terminologyMisc)*)            >

<!ATTLIST ofig                %a.global
          type                CDATA               #IMPLIED      >
```

Discussed in: 13.4.1 ('DTD Fragment for Nested Style') on p. 321; 13.2 ('Tags for Terminological Data') on p. 312

opener

<opener> groups together dateline, byline, salutation, and similar phrases appearing as a preliminary group at the start of a division, especially of a letter.

Attributes: [None: global and inherited attributes only.]
Example:

```
<opener><dateline>Walden, this 29. of August 1592</dateline></opener>
```

Example:

```
 <opener>
   <dateline><place>Great Marlborough Street</place>
   <date>November 11, 1848</date>
   </dateline>
   <salute>My dear Sir,</salute>
</opener>
<p>I am sorry to say that absence from town and other
circumstances have prevented me from earlier enquiring...
```

Part: base tag set for common core features
Member of classes: divtop
DTD file: teistr2
Occurs within: back body castList div div0 div1 div2 div3 div4 div5 div6 div7 epilogue front group lg performance prologue
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateline dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s salute seg sic signed soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT opener        - O  (signed | dateline | salute |
                              %phrase.seq;)*                    >

<!ATTLIST opener              %a.global                         >
```

Discussed in: 7.2.4 ('Content of Textual Divisions') on p. 194; 7.2 ('Elements Common to All Divisions') on p. 190

oRef

<oRef> (i.e. orthographic-form reference) in a dictionary example, indicates a reference to the orthographic form(s) of the headword.

Attributes:

type indicates the kind of typographic modification made to the headword in the reference.
Data type: CDATA
Sample values include:
cap indicates first letter is given as capital
nohyph indicates that the headword, though a prefix or suffix, loses its hyphen
Default: #IMPLIED

Part: base tag set for printed dictionaries
Member of classes: dictionaries, formPointers [and indirectly also:] phrase
DTD file: teidict2
Data description: Empty element.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm oVar p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: [none]
Declaration:

```
<!ELEMENT oRef          - O  EMPTY                              >
<!ATTLIST oRef                %a.global
                              %a.dictionaries
                              %a.formPointers
          type                CDATA                #IMPLIED     >
```

Discussed in: 12.4 ('Headword and Pronunciation References') on p. 297

orgdivn

<orgdivn> (i.e. organizational division) indicates a division, branch or department specified in an organizational name.

Attributes:

type more fully describes the organization division specified in the name component. Possible values include "branch", "department", "section", "division", etc.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED

reg (i.e. regularization) gives a normalized or regularized form of the organizational division.
Data type: CDATA
Value: Any string of characters.
Default: #IMPLIED
In providing a "regularized" form, no claim is made that the form in the source text is incorrect; the regularized form is simply that chosen as the main form for purposes of unifying variant forms under a single heading.

Example:

```
<orgname type=academic key=DMHGU1>
<orgdivn type=department>
    Department of
    <orgtype type=function>Modern History</orgtype>
</orgdivn>,
Glasgow
<orgtype type=function>University</orgtype></orgname>
```

Part: additional tag set for
Member of classes:
DTD file: teind2
Occurs within: [none]
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT orgdivn       - -  (%phrase.seq)                    >
<!ATTLIST orgdivn            %a.global
          type               CDATA              #IMPLIED
          reg                CDATA              #IMPLIED      >
```

Discussed in: 20.3 ('Organization Names') on p. 497

<orgName> (i.e. organization name) contains an organizational name. orgName

Attributes:

type more fully describes the organization indicated in the organizational name. Possible values include "voluntary", "political", "governmental", "industrial", "commercial", etc.
Data type: CDATA
Value:
Default: #IMPLIED

key provides an alternative identifier for the organization being named, such as a database record key.
Data type: CDATA
Value: any string
Default: #IMPLIED
The value may be a unique identifier from a database, or simply a more explicit name for the referent. Its purpose is only to record an identification; if the analysis leading to the identification is to be recorded as well, the analytic tags described in chapter 16 ('Feature Structures') on p. 397 should be used in addition or instead.

reg (i.e. regularization) gives a normalized or regularized form of the organization name
Data type: CDATA
Value: Any string of characters.
Default: #IMPLIED
In providing a "regularized" form, no claim is made that the form in the source text is incorrect; the regularized form is simply that chosen as the main form for purposes of unifying variant forms under a single heading.

Example:

```
About a year back, a question of considerable interest was
agitated in the
<orgname key=PAS1 type=voluntary reg=Pennsylvania Abolition Society'>
Pennsyla. Abolition Society</orgname>.
```

Part: additional tag set for
Member of classes:
DTD file: teind2
Occurs within: [none]
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT orgName       - -  (%phrase.seq)                    >
<!ATTLIST orgName            %a.global
          type               CDATA              #IMPLIED
          key                CDATA              #IMPLIED
          reg                CDATA              #IMPLIED      >
```

Discussed in: 20.3 ('Organization Names') on p. 497

orgtitle

<orgtitle> (i.e. organization title) contains the proper name component of an organizational name.

Attributes:

type more fully describes the organization title. Possible values include "formal", "colloquial", "acronym", etc.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED

reg (i.e. regularization) gives a normalized or regularized form of the organization title.
Data type: CDATA
Value: Any string of characters.
Default: #IMPLIED
In providing a "regularized" form, no claim is made that the form in the source text is incorrect; the regularized form is simply that chosen as the main form for purposes of unifying variant forms under a single heading.

Example:

```
Mr Frost will be able to earn an extra fee from
<orgname type=media key=BSB1>
<orgtitle type=acronym>BSkyB</orgtitle></orgname>
rather than the <orgname type=media key=BBC1>
<orgtitle type=acronym reg='British Broadcasting Corporation'>BBC
</orgtitle></orgname>
```

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teind2
Occurs within: [none]
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT orgtitle      - -  (%phrase.seq)                    >

<!ATTLIST orgtitle           %a.global
          type               CDATA              #IMPLIED
          reg                CDATA              #IMPLIED      >
```

Discussed in: 20.3 ('Organization Names') on p. 497

orgtype

<orgtype> (i.e. organization type) indicates a part of the organization name which contains information about the organization's structure or function.

Attributes:

type more fully describes the organization type specified in the name component. Possible values include "function", "structure", etc.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED

reg (i.e. regularization) gives a normalized or regularized form of the organization type
Data type: CDATA
Value: Any string of characters.
Default: #IMPLIED
In providing a "regularized" form, no claim is made that the form in the source text is incorrect; the regularized form is simply that chosen as the main form for purposes of unifying variant forms under a single heading.

Example:

```
<orgname type='utility company' key=WWPC1>Washington
<orgtype type=function>Water Power</orgtype> <orgtype
reg='incorporated' type=structure>Inc.</orgtype></orgname>
```

Part: additional tag set for
Member of classes:
DTD file: teind2
Occurs within: [none]
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT orgtype       - -  (%phrase.seq)                    >

<!ATTLIST orgtype            %a.global
          type               CDATA              #IMPLIED
          reg                CDATA              #IMPLIED      >
```

Discussed in: 20.3 ('Organization Names') on p. 497

orig

<orig> (i.e. original form) contains the original form of a reading, for which a regularized form is given in an attribute value.

Attributes:

reg (i.e. original) gives a regularized (normalized) form of the text.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED

resp (i.e. responsibility) identifies the individual responsible for the regularization of the word or phrase.
Data type: CDATA
Value: any string of characters, typically the initials of the individual involved, or a role identifier like "editor" if not known by name.
Default: #IMPLIED

Part: additional tag set for common core features
Member of classes: edit [and indirectly also:] phrase
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT orig          - -  (%phrase.seq;)                   >

<!ATTLIST orig               %a.global
          reg                CDATA              #IMPLIED
          resp               CDATA              #IMPLIED      >
```

Discussed in: 6.5.2 ('Regularization and Normalization') on p. 143; 19 ('Critical Apparatus') on p. 467

orth

<orth> (i.e. orthographic form) gives the orthographic form of a dictionary headword.

Attributes:

type gives the type of spelling.
Data type: CDATA
Value: Any convenient word or phrase, e.g. "lat" (latinate), "std" (standard), "trans" (transliterated), etc.
Default: #IMPLIED

extent gives the extent of the orthographic information provided.
Data type: CDATA
Sample values include:

full full form
pref prefix
suff suffix
part partial

Default: FULL

Example:

```
<!>
```

Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, formInfo
DTD file: teidict2
Data description: May contain character data and phrase-level elements.
Occurs within: form trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT orth          - O  (%paraContent)                    >

<!ATTLIST orth               %a.global
                             %a.dictionaries
          type               CDATA              #IMPLIED
          extent             CDATA              full          >
```

Discussed in: 12.3.1 ('Information on Written and Spoken Forms') on p. 279

otherForm

<otherForm> (i.e. other form) contains an alternate designation for the concept treated by the term entry, such as a synonym.

Attributes:

type classifies the <otherForm> using some typology, preferably the dictionary of data element types specified in ISO WD 12 620.
Data type: CDATA
Value: any string of characters; for serious terminological work, values should be taken from the dictionary of data element types specified in ISO WD 12 620.
Default: #IMPLIED

In nested term entries, the <otherForm> belongs to the same terminological information group as the term for which it is an alternate.

Part: base tag set for terminological data
Member of classes:
DTD file: teite2n teite2f
Data description: May contain character data and phrase-level elements.
Occurs within: ofig

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref

Declaration:

```
<!ELEMENT otherForm     - O  (%paraContent;)                  >
<!ATTLIST otherForm          %a.global
          type               CDATA               #IMPLIED    >
```

Discussed in: 13.4.2 ('DTD Fragment for Flat Style') on p. 322; 13.4.1 ('DTD Fragment for Nested Style') on p. 321; 13.2 ('Tags for Terminological Data') on p. 312

<oVar> (i.e. orthographic-variant reference) in a dictionary example, indicates a reference to variant orthographic form(s) of the headword. oVar

Attributes:

type indicates the kind of variant involved.
Data type: CDATA
Sample values include:

pt past tense
pp past participle
prp present participle
f feminine
pl plural

Default: #IMPLIED

Part: base tag set for printed dictionaries
Member of classes: dictionaries, formPointers [and indirectly also:] phrase
DTD file: teidict2
Data description: Character data or <oRef>.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA oRef
Declaration:

```
<!ELEMENT oVar          - -  (#PCDATA | oRef)*                >
<!ATTLIST oVar               %a.global
                             %a.dictionaries
                             %a.formPointers
          type               CDATA               #IMPLIED    >
```

Discussed in: 12 ('Print Dictionaries') on p. 269

<p> (i.e. paragraph) marks paragraphs in prose. p

Attributes: [None: global and inherited attributes only.]
Example:

```
<p>Hallgerd was outside.  <q>There is blood on your
axe,</q> she said.  <q>What have you done?</q></p>
<p><q>I have now arranged that you can be married a
```

```
second time,</q> replied Thjostolf.</p>
<p><q>Then you must mean that Thorvald is dead,</q>
she said.</p>
<p><q>Yes,</q> said Thjostolf.  <q>And now you must
think up some plan for me.</q>
```

In some contexts, the paragraph may have a specialized meaning, e.g. in the tag set for dictionaries, <p> is used to enclose any running text, and thus does not imply text set off as is conventionally done in running prose.
Part: additional tag set for common core features
Member of classes: chunk [and indirectly also:] common
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: add argument availability body broadcast castList corr correction div div0 div1 div2 div3 div4 div5 div6 div7 editionStmt editorialDecl encodingDesc epigraph epilogue equipment equiv exemplum figure hyphenation interpretation item langUsage metDecl normalization note particDesc particLinks performance person personGrp projectDesc prologue publicationStmt q quotation quote recording recordingStmt refsDecl remarks samplingDecl scriptStmt segmentation seriesStmt set setting settingDesc sic sourceDesc sp stage stdVals view
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT p              - O  (%paraContent;)                >
<!ATTLIST p                   %a.global                      >
```

Discussed in: 6.1 (‘Paragraphs’) on p. 120; 10.2.4 (‘Speech Contents’) on p. 241

page

<page> marks pages in a reference edition.

Attributes: [None: global and inherited attributes only.]
Example:

```
<(La)page n=32> ... [text of edition La, page 32]
<(La)page n=32> ... [text of edition La, page 33]
<(La)page n=32> ... [text of edition La, page 34]
<(La)page n=32> ... [text of edition La, page 35]
```

The <page> element in a concurrent page-reference DTD may be subdivided into columns or lines if desired, or left unanalysed.
Part: base tag set for
Member of classes:
DTD file: teipl2
Data description: May contain <col> elements, <line> elements, or character data.
Occurs within: %version vol
May contain: #PCDATA col line
Declaration:

```
<!ELEMENT page           - O  (#PCDATA | line | col)*        >
<!ATTLIST page                %a.global                      >
```

Discussed in: 31.6 (‘Concurrent Markup for Pages and Lines’) on p. 637

parents

<parents> lists elements which can directly contain this element.

Attributes: [None: global and inherited attributes only.]
Example:

```
<parents>head, title
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2

Data description: a list of valid SGML names separated by commas or spaces.
Occurs within: tagDoc
May contain: #PCDATA
Declaration:

```
<!ELEMENT parents       - O  (#PCDATA)                          >
<!ATTLIST parents            %a.global                          >
```

Discussed in: 27.1 ('The TagDoc Documentation Element') on p. 603

part

<part> specifies the module or part to which a particular element, element class, or entity belongs in a modular encoding scheme such as the TEI.

Attributes:

type indicates whether the tag set is a base, additional, core, or auxiliary tag set.
Data type: CDATA
Sample values include:
core a core tag set (part of every document)
base a base tag set
add an additional tag set
aux an auxiliary tag set
Default: #IMPLIED

name indicates the specific tag set or part in question, usually by means of an identifier or short form.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED

Example:

```
<part type=base name=di>Base tag set for print dictionaries</part>
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: A descriptive phrase identifying the module concerned.
Occurs within: classDoc tagDoc
May contain: #PCDATA
Declaration:

```
<!ELEMENT part          - O  (#PCDATA)                          >
<!ATTLIST part               %a.global
          type               CDATA               #IMPLIED
          name               CDATA               #IMPLIED     >
```

Discussed in: 27.1 ('The TagDoc Documentation Element') on p. 603; 27.2 ('Element Classes') on p. 606

particDesc

<particDesc> (i.e. participation description) describes the identifiable speakers, voices or other participants in a linguistic interaction.

Attributes: [None: global and inherited attributes only.]
Example:

```
<particDesc>
<participant id=P1 sex=F age=42>
  <p>Female informant, well-educated, born in Shropshire
   UK, 12 Jan 1950, of unknown occupation.
   Speaks French fluently. Socio-Economic status B2.
</participant>
<participant id=P2 sex=M age=43>
<particLinks>
   <relation active="P1 P2" desc="spouse">
</particLinks>
</particDesc>
```

Part: auxiliary tag set for corpora and collections
Member of classes: declarable
DTD file: teicorp2
Data description: May contain a prose description organized as paragraphs, or a structured list of participants and participant groups, with an optional formal specification of any relationships amongst them.
Occurs within: profileDesc
May contain: p particLinks person personGrp
Declaration:

```
<!ELEMENT particDesc    - O  (p+ | ( (person | personGrp)+,
                              particLinks?) )                    >

<!ATTLIST particDesc         %a.global
                             %a.declarable                       >
```

Discussed in: 23.2 ('Contextual Information') on p. 540; 5.4 ('The Profile Description') on p. 108; 23.3.2 ('Declarable Elements') on p. 552

particLinks

<particLinks> (i.e. participant relationships) describes the relationships or social links existing between participants in a linguistic interaction.

Attributes: [None: global and inherited attributes only.]
Example:

```
<particLinks>
  <relation desc='parent' active="P1 P2"
         passive="P3 P4" mutual=N>
  <relation desc='spouse' active="P1 P2">
  <relation class='social' desc='employer'
          active=P1 passive='P3 P5 P6 P7' mutual=N>
</particLinks>
```

Part: auxiliary tag set for corpora and collections
Member of classes:
DTD file: teicorp2
Data description: May contain a prose description organized as paragraphs, or a sequence of <relation> elements.
Occurs within: particDesc
May contain: p relation
Declaration:

```
<!ELEMENT particLinks   - O  (p+ | relation+)                    >

<!ATTLIST particLinks        %a.global                           >
```

Discussed in: 23.2.2 ('The Participants Description') on p. 545

pause

<pause> a pause either between or within utterances.

Attributes:

type categorizes the pause in some respect.
Data type: CDATA
Value: An open list
Default: #IMPLIED

who supplies an identifier for the person or group pausing. Its value is the identifier of a <participant> or <participant.grp> element in the TEI header.
Data type: IDREF
Value: Must identify a participant or participant group within the TEI Header
Default: #IMPLIED

Example:

```
<pause type=pregnant dur=42>
```

Part: base tag set for spoken materials
Member of classes: comp.spoken, timed

DTD file: teispok2
Data description: empty
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage u view
May contain: [none]
Declaration:

```
<!ELEMENT pause          - O  EMPTY                              >

<!ATTLIST pause               %a.global
                              %a.timed
          type                CDATA                #IMPLIED
          who                 IDREF                #IMPLIED       >
```

Discussed in: 11.2.7 ('Formal Definition') on p. 259; 11.2 ('Elements Unique to Spoken Texts') on p. 253; 11.2.2 ('Pause') on p. 255

<pb> (i.e. page break) marks the boundary between one page of a text and the next in a standard reference system. pb

Attributes:

ed (i.e. edition) indicates the edition or version in which the page break is located at this point
Data type: CDATA
Value: Any string of characters; usually a siglum conventionally used for the edition.
Default: #IMPLIED
Example:

```
<pb ed=Riverside n=123>
```

n (i.e. number or name) indicates the number or other value associated with the page which follows the point of insertion of this <pb>.
Data type: CDATA
Value: Any string of characters.
Default: #IMPLIED
Encoders should adopt a clear and consistent policy as to whether the numbers associated with page breaks relate to the physical sequence number of the page or the page number or signature printed on it. By convention, <pb> elements should appear at the start of the page to which they refer.

Like other forms of milestone tag, <pb> tags cannot be automatically verified by SGML; for better validation, a concurrent markup stream should be used.

Part: additional tag set for common core features
Member of classes: refsys
DTD file: teicore2
Occurs within: [none]
May contain: [none]
Declaration:

```
<!ELEMENT pb             - O  EMPTY                              >

<!ATTLIST pb                  %a.analysis
                              %a.linking
                              %a.terminology
          id                  ID                   #IMPLIED
          lang                IDREF                %INHERITED

          rend                CDATA                #IMPLIED
          ed                  CDATA                #IMPLIED
          n                   CDATA                #IMPLIED       >
```

Discussed in: 6.9.3 ('Milestone Tags') on p. 158

<per> (i.e. person) contains an indication of the grammatical person (1st, 2d, 3d, etc.) associated with a given inflected form in a dictionary. per

Attributes: [None: global and inherited attributes only.]
This element is synonymous with <gram type=person>.
Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, morphInfo
DTD file: teidict2
Data description: May contain character data and phrase-level elements.
Occurs within: etym trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT per           - O  (%paraContent;)                >

<!ATTLIST per                %a.global
                             %a.dictionaries                >
```

Discussed in: 12.3.1 ('Information on Written and Spoken Forms') on p. 279

performance

<performance> contains a section of front or back matter describing how a dramatic piece is to be performed in general or how it was performed on some specific occasion.

Attributes: [None: global and inherited attributes only.]
Example:

```
<performance>
<p><rs type=place>Gateway Theatre, Edinburgh</rs>,
<date>6 September 1948</date>
<castList>
    <castitem><role>Anath Bithiah</>
       <actor>Athene Seyler</actor>
    <!-- ... -->
    <castitem><role>Shendi</role>
       <actor>Robert Rietty</actor>
  </castList>
  <p>Directed by <name>E. Martin Browne</name>
</performance>
```

Example:

```
<performance>
  <p>Cast of the original production at the
    <rs type=place>Savoy Theatre, London,</rs>
  on <date>September 24, 1907</>
  <castList>
    <castitem>Colonel Hope : Mr A.E.George</>
  ...
  </castList>
</performance>
```

Part: base tag set for performance texts
Member of classes: dramafront [and indirectly also:] front
DTD file: teidram2
Data description: contains paragraphs and an optional cast list only.
Occurs within: back front
May contain: argument bibl biblFull biblStruct byline camera caption castList cit closer docAuthor docDate entry entryFree epigraph event eTree graph head kinesic l label lg lg1 list listBibl move note opener p pause q quote salute shift signed sound sp stage superentry tech termEntry trailer tree u view vocal writing
Declaration:

```
<!ELEMENT performance   - -  ((%m.divtop)*, (%component)+,
                             (%m.divbot)*)                  >

<!ATTLIST performance        %a.global                      >
```

Discussed in: 10.1.3 (‘Records of Performances’) on p. 232; 10.1 (‘Front and Back Matter ’) on p. 228

<persName> (i.e. personal name) contains a proper noun or proper-noun phrase referring to a person, possibly including any or all of the person’s forename, surname, honorofic, added names, etc. persName

Attributes:

type describes the personal name more fully using an open-ended list of words or phrases which help to indicate the function, e.g. “married name”, “maiden name”, “pen name”, “religious name”, etc.
Data type: CDATA
Value: Any string of characters.
Default: #IMPLIED

Example:

```
<persName>
   <forename>Edward</forename>
   <forename>George</forename>
   <surname type=linked>Bulwer-Lytton</surname>,
   <roleName>Baron Lytton of
     <placeName>Knebworth</placeName>
   </roleName>
</persName>
```

Part: additional tag set for names and dates
Member of classes: demographic, names
DTD file: teind2
Occurs within: person personGrp
May contain: #PCDATA abbr add address addName anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign forename formula gap genName gi gloss handShift hi lang link m measure mentioned name nameLink num orig oRef oVar phr ptr pRef pVar ref reg roleName rs s seg sic soCalled surname tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT persName      - -  (%m.personPart; | %m.phrase; |
                              #PCDATA)*                      >
<!ATTLIST persName           %a.global
                             %a.names
          type               CDATA              #IMPLIED    >
```

Discussed in: 20.1 (‘Personal Names’) on p. 488

<person> describes a single participant in a language interaction. person

Attributes:

role specifies the role of this participant in the group.
Data type: CDATA
Value: a set of keywords to be defined
Default: #IMPLIED

sex specifies the sex of the participant.
Data type: (M | F | U)
Sample values include:

m male
f female
u unknown or inapplicable

Default: #IMPLIED

age specifies the age group to which the participant belongs.
Data type: CDATA
Value: suggested values are to be supplied
Default: #IMPLIED

Example:

```
<person id=P1 sex=F age=42>
  <p>Female informant, well-educated, born in Shropshire
   UK, 12 Jan 1950, of unknown occupation.
   Speaks French fluently. Socio-Economic status B2.
</person>
```

Part: auxiliary tag set for corpora and collections
Member of classes:
DTD file: teicorp2
Data description: May contain a prose description organized as paragraphs, or any sequence of demographic elements in any combination.
Occurs within: particDesc
May contain: birth education firstLang occupation p persName residence socecStatus
Declaration:

```
<!ELEMENT person        - O  (p+ | (%m.demographic)* )          >

<!ATTLIST person            %a.global
          role              CDATA               #IMPLIED
          sex               (m | f | u )        #IMPLIED
          age               CDATA               #IMPLIED      >
```

Discussed in: 23.2.2 ('The Participants Description') on p. 545

personGrp

<personGrp> (i.e. personal group) describes a group of individuals treated as a single person for analytic purposes.

Attributes:

role specifies the role of this group of participants in the interaction.
Data type: CDATA
Value: a set of keywords to be defined
Default: #IMPLIED

sex specifies the sex of the participant group.
Data type: (M | F | U | X)
Sample values include:
m male
f female
u unknown
x mixed
Default: #IMPLIED

age specifies the age group of the participants.
Data type: CDATA
Value: suggested values are to be supplied
Default: #IMPLIED

size specifies the size or approximate size of the group.
Data type: CDATA
Value: may contain a number and an indication of accuracy, e.g. 'approx 200'
Default: #IMPLIED

Example:

```
<particGrp role=audience sex=x size='approx 50' id=PG1>
```

The global id attribute should be used to identify each speaking participant in a spoken text if the who attribute is specified on individual utterances.
Part: auxiliary tag set for corpora and collections
Member of classes:
DTD file: teicorp2
Data description: May contain a prose description organized as paragraphs, or any sequence of demographic elements in any combination.
Occurs within: particDesc

May contain: birth education firstLang occupation p persName residence socecStatus
Declaration:

```
<!ELEMENT personGrp     - O  (p+ | (%m.demographic)* )          >

<!ATTLIST personGrp          %a.global
          role               CDATA               #IMPLIED
          sex                (m | f | u | x)     #IMPLIED
          age                CDATA               #IMPLIED
          size               CDATA               #IMPLIED       >
```

Discussed in: 23.2.2 ('The Participants Description') on p. 545

<phr> (i.e. phrase) represents a grammatical phrase. phr

Attributes: [None: global and inherited attributes only.]
Example:

```
<phr type=verb function='extraposted modifier'>To talk
  <phr type=preposition function=complement>of
    <phr type=noun function=object>many things</phr>
  </phr>
</phr>
```

Part: additional tag set for
Member of classes: seg [and indirectly also:] phrase
DTD file: teiana2
Data description: May contain character data and phrase-level elements.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT phr           - -  (%phrase.seq)                     >

<!ATTLIST phr                %a.global
                             %a.seg                         >
```

Discussed in: 15.1 ('Linguistic Segment Categories') on p. 382

<placeName> (i.e. place name) contains an absolute or relative place name. placeName

Attributes: [None: global and inherited attributes only.]
Example:

```
<placeName>
  <settle>Rochester</settle>
  <region>New York</region>
</placeName>

<placeName>
  <geogName>Arrochar Alps</geogName>
  <region>Argylshire</region>
</placeName>
```

```
<placeName>
  <distance>10 miles</distance>
  <offset>Northeast of</offset>
  <settle>Attica</settle>
</placeName>
```

Part: auxiliary tag set for names and dates
Member of classes: names, placePart
DTD file: teind2
Occurs within: placeName
May contain: #PCDATA abbr add address anchor att bloc c caesura cl corr country date dateRange dateStruct del distance distinct emph expan foreign formula gap geog geogName gi gloss handShift hi lang link m measure mentioned name num offset orig oRef oVar phr placeName ptr pRef pVar ref reg region rs s seg settlement sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT placeName     - -  ((%m.placePart; | %m.phrase; |
                              #PCDATA)+)                    >

<!ATTLIST placeName           %a.global
          type                CDATA               #IMPLIED
          full                (yes | abb | init)  yes
                              %a.names                      >
```

Discussed in: 20.2 ('Place Names') on p. 493

plus

<plus> (i.e. Binary plus value) provides binary plus value for a feature.

Attributes: [None: global and inherited attributes only.]
Example:

```
<f name=strident><plus></f>
```

Part: additional tag set for feature structures
Member of classes: binary [and indirectly also:] singleVal
DTD file: teifs2
Data description: Empty element.
Occurs within: f fvLib vAlt
May contain: [none]
Declaration:

```
<!ELEMENT plus          - O  EMPTY                        >

<!ATTLIST plus                %a.global                   >
```

Discussed in: 16.2 ('Elementary Feature Structures: Features with Binary Values') on p. 398

pos

<pos> (i.e. part of speech) indicates the part of speech assigned to a dictionary headword (noun, verb, adjective, etc.)

Attributes: [None: global and inherited attributes only.]
Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, gramInfo
DTD file: teidict2
Data description: May contain character data and phrase-level elements.
Occurs within: gramGrp trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT pos           - O  (%paraContent;)              >
```

```
<!ATTLIST pos                  %a.global
                               %a.dictionaries                 >
```

Discussed in: 12.3.2 (‘Grammatical Information’) on p. 284

<postBox> (i.e. postBox) contains a number or other identifier for some postal delivery point other than a street address. postBox

Attributes: [None: global and inherited attributes only.]
Example:

```
<postBox>P.O. Box 280
```

Example:

```
<postBox>Postbus 532
```

The position and nature of postal codes is highly country-specific; the conventions appropriate to the country concerned should be used.
Part: base tag set for common core features
Member of classes: addrPart
DTD file: teicore2
Occurs within: address
May contain: #PCDATA
Declaration:

```
<!ELEMENT postBox        - O  (#PCDATA)                        >
<!ATTLIST postBox              %a.global                       >
```

Discussed in: 6.4.2 (‘Addresses’) on p. 134

<postCode> (i.e. postCode) contains a numerical or alphanumeric code used as part of a postal address to simplify sorting or delivery of mail. postCode

Attributes: [None: global and inherited attributes only.]
Example:

```
<postCode>HR1 3LR
```

Example:

```
<postCode>60142-7
```

The position and nature of postal codes is highly country-specific; the conventions appropriate to the country concerned should be used.
Part: base tag set for common core features
Member of classes: addrPart
DTD file: teicore2
Occurs within: address
May contain: #PCDATA
Declaration:

```
<!ELEMENT postCode       - O  (#PCDATA)                        >
<!ATTLIST postCode             %a.global                       >
```

Discussed in: 6.4.2 (‘Addresses’) on p. 134

<pRef> (i.e. pronunciation reference) in a dictionary example, indicates a reference to the pronunciation(s) of the headword. pRef

Attributes: [None: global and inherited attributes only.]
Part: base tag set for printed dictionaries
Member of classes: dictionaries, formPointers [and indirectly also:] phrase
DTD file: teidict2
Data description: Empty element.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor

docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose pVar q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: [none]

Declaration:

```
<!ELEMENT pRef          - O  EMPTY                              >
<!ATTLIST pRef               %a.global
                             %a.dictionaries
                             %a.formPointers                    >
```

Discussed in: 12.4 ('Headword and Pronunciation References') on p. 297

preparedness

<preparedness> describes the extent to which a text may be regarded as prepared or spontaneous.

Attributes:

type a keyword characterizing the type of preparedness.
Data type: CDATA
Sample values include:
none spontaneous or unprepared
scripted follows a script
formulaic follows a predefined set of conventions
revised polished or revised before presentation
Default: #IMPLIED

Example:

```
<preparedness type='none'>
```

Part: auxiliary tag set for corpora and collections
Member of classes:
DTD file: teicorp2
Data description:
Occurs within: textDesc
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT preparedness  - O  (%phrase.seq)                      >
<!ATTLIST preparedness       %a.global
          type               CDATA              #IMPLIED       >
```

Discussed in: 23.2.1 ('The Text Description') on p. 541

principal

<principal> (i.e. principal researcher) supplies the name of the principal researcher responsible for the creation of an electronic text.

Attributes: [None: global and inherited attributes only.]

Example:

```
<principal>Gary Taylor
```

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:

Occurs within: titleStmt
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT principal     - O  (%phrase.seq;)                    >
<!ATTLIST principal          %a.global                         >
```

Discussed in: 5.2.1 ('The Title Statement') on p. 82

profileDesc

<profileDesc> (i.e. text-profile description) provides a detailed description of non-bibliographic aspects of a text, specifically the languages and sublanguages used, the situation in which it was produced, the participants and their setting.

Attributes: [None: global and inherited attributes only.]
Example:

```
<profileDesc>
  <langUsage><language wsd=FR>French
  <textDesc n=novel>
    <channel mode=w>print; part issues
    <constitution type=single>
    <derivation type=original>
    <domain type=art><factuality type=fiction>
    <interaction type=none>
    <preparedness type=prepared>
    <purposes><purpose type=entertain degree=high>
              <purpose type=inform degree=medium>
    </purposes>
  </textDesc>
  <settingDesc>
    <setting><place>Paris, France
             <time>Late 19th century
    </setting>
  </settingDesc>
</profileDesc>
```

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:
Occurs within: teiHeader
May contain: creation handList langUsage particDesc settingDesc textClass textDesc
Declaration:

```
<!ELEMENT profileDesc   - -  (creation?, langUsage*, textDesc*,
                              particDesc*, settingDesc*,
                              handList*, textClass*)            >
<!ATTLIST profileDesc        %a.global                          >
```

Discussed in: 5.4 ('The Profile Description') on p. 108; 5.1.1 ('The TEI Header and Its Components') on p. 78

projectDesc

<projectDesc> (i.e. project description) describes in detail the aim or purpose for which an electronic file was encoded, together with any other relevant information concerning the process by which it was assembled or collected.

Attributes: [None: global and inherited attributes only.]
Example:

```
<projectDesc><p>Texts collected for use in the Claremont Shakespeare
  Clinic, June 1990</projectDesc>
```

Part: auxiliary tag set for TEI headers

Member of classes: declarable
DTD file: teihdr2
Data description:
Occurs within: encodingDesc
May contain: p
Declaration:

```
<!ELEMENT projectDesc   - O  (p+)                          >

<!ATTLIST projectDesc        %a.global
                             %a.declarable                 >
```

Discussed in: 5.3.1 ('The Project Description') on p. 95; 5.3 ('The Encoding Description') on p. 93; 23.3.2 ('Declarable Elements') on p. 552

prologue

<prologue> contains the prologue to a drama, typically spoken by an actor out of character, possibly in association with a particular performance or venue.

Attributes: [None: global and inherited attributes only.]
Example:

```
<prologue>
<sp>
<l>Wits, like physicians never can agree,
<l>When of a different society.
<!-- ... -->
<l>New plays are stuffed with wits, and with deboches,
<l>That crowd and sweat like cits in May-Day coaches.
</sp>
<trailer>
Written by a person of quality
</trailer>
</prologue>
```

Part: base tag set for performance texts
Member of classes: dramafront [and indirectly also:] front
DTD file: teidram2
Data description:
Occurs within: back front
May contain: argument bibl biblFull biblStruct byline camera caption castList cit closer docAuthor docDate entry entryFree epigraph event eTree graph head kinesic l label lg lg1 list listBibl move note opener p pause q quote salute shift signed sound sp stage superentry tech termEntry trailer tree u view vocal writing
Declaration:

```
<!ELEMENT prologue      - -  ((%m.divtop)*, (%component)+,
                              (%m.divbot)*)                >

<!ATTLIST prologue           %a.global                     >
```

Discussed in: 10.1.2 ('Prologues and Epilogues') on p. 230; 10.1 ('Front and Back Matter ') on p. 228

pron

<pron> (i.e. pronunciation) contains the pronunciation(s) of the word.

Attributes:

extent indicates whether the pronunciation is for whole word or part.
Data type: CDATA
Sample values include:
full full form
pref prefix
suff suffix
part partial
Default: FULL

notation indicates what notation is used for the pronunciation, if more than one occurs in the machine-readable dictionary.
Data type: CDATA
Value: Sample values: IPA, Murray, ...
Default: #IMPLIED

Example:

```
<entry>
    <form>
         <orth>obverse</orth>
         <pron>&ACCENT;&auml;b-&accent;v&schwa;rs</pron>,
         <pron extent=prefix>&auml;b-&ACCENT;</pron>,
         <pron extent=prefix>&schwa;b-&ACCENT;</pron>
    </form>
    <gramGrp><pos>n</pos></gramGrp>
    <!-- ... -->
</entry>
(From:  Webster's Collegiate, obverse 2)
```

Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, formInfo
DTD file: teidict2
Data description: May contain character data and phrase-level elements.
Occurs within: form trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref

Declaration:

```
<!ELEMENT pron          - O  (%paraContent)                  >

<!ATTLIST pron               %a.global
                             %a.dictionaries
          extent             CDATA               full
          notation           CDATA               #IMPLIED     >
```

Discussed in: 12.3.1 ('Information on Written and Spoken Forms') on p. 279

<ptr> defines a pointer to another location in the current document in terms of one or more identifiable elements. ptr

Attributes:

target specifies the destination of the pointer as one or more SGML identifiers
Data type: IDREFS
Value: Each value specified must be the same as that specified as value for an ID attribute for some other element in the current SGML document.
Default: #REQUIRED

Example:

```
<ptr target="P2 P5" resp=LB type=s>
<ptr target='D1' type='defn'>
```

Part: base tag set for common core features
Member of classes: loc, pointer, terminologyInclusions [and indirectly also:] phrase
DTD file: teicore2
Occurs within: abbr activity actor add addrLine addName admin affiliation altGrp author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cit cl classCode classDoc closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph entDoc equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype joinGrp l label lang language langKnown lbl lem linkGrp locale measure meeting mentioned mood name nameLink note num number occasion

occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagDoc tagUsage tech term termEntry time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: [none]

Declaration:

```
<!ELEMENT ptr           - O  EMPTY                              >
<!ATTLIST ptr                %a.global
                             %a.pointer
          target             IDREFS               #REQUIRED    >
```

Discussed in: 6.6 ('Simple Links and Cross References') on p. 147; 14.1 ('Pointers') on p. 333

publicationStmt

<publicationStmt> (i.e. publication statement) groups information concerning the publication or distribution of an electronic or other text.

Attributes: [None: global and inherited attributes only.]

Example:

```
<publicationStmt>
<publisher>C. Muquardt <pubPlace>Bruxelles & Leipzig
<date>1846
```

Example:

```
<publicationStmt>
<publisher>Chadwyck Healey
<pubPlace>Cambridge
<availability>Available under licence only
<date>1992
```

Part: auxiliary tag set for TEI headers

Member of classes:

DTD file: teihdr2

Data description:

Occurs within: biblFull fileDesc

May contain: address authority availability date distributor idno p publisher pubPlace

Declaration:

```
<!ELEMENT publicationStmt
                         - O  (p+ | ( (publisher | distributor |
                              authority) & (pubPlace?, address?,
                              idno*, availability?, date?)+ )+ )
                                                              >
<!ATTLIST publicationStmt    %a.global                        >
```

Discussed in: 5.2.4 ('Publication, Distribution, etc') on p. 86; 5.2 ('The File Description') on p. 80

publisher

<publisher> provides the name of the organization responsible for the publication or distribution of a bibliographic item.

Attributes: [None: global and inherited attributes only.]

Example:

```
<imprint><pubPlace>Oxford</pubPlace>
        <publisher>Clarendon Press</publisher>
        <date>1987</date>
</imprint>
```

Part: base tag set for common core features

Member of classes: biblPart

DTD file: teicore2

Data description: Use the full form of the name by which a company is usually referred to, rather than any abbreviation of it which may appear on a title page

Occurs within: bibl docImprint imprint publicationStmt
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT publisher     - O  (%phrase.seq)                    >
<!ATTLIST publisher          %a.global                        >
```

Discussed in: 6.10.2.3 ('Imprint, Pagination, and Other Details') on p. 171; 5.2.4 ('Publication, Distribution, etc') on p. 86

<pubPlace> contains the name of the place where a bibliographic item was published. pubPlace

Attributes: [None: global and inherited attributes only.]
Part: additional tag set for common core features
Member of classes: biblPart, names
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: bibl docImprint imprint publicationStmt
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT pubPlace      - -  (%phrase.seq;)                   >
<!ATTLIST pubPlace           %a.global
                             %a.names                         >
```

Discussed in: 6.10.2.3 ('Imprint, Pagination, and Other Details') on p. 171

<purpose> characterizes a single purpose or communicative function of the text. purpose

Attributes:

type specifies a particular kind of purpose.
Data type: CDATA
Sample values include:

persuade didactic, advertising, propaganda, etc.
express self expression, confessional, etc.
inform convey information, educate, etc.
entertain amuse, entertain, etc.

Default: #IMPLIED

degree specifies the extent to which this purpose predominates.
Data type: (HIGH | MEDIUM | LOW | UNKNOWN)
Sample values include:

high this purpose is predominant
medium this purpose is intermediate
low this purpose is weak
unknown extent unknown

Default: #IMPLIED

Example:

```
<purpose type=persuade degree=high>
<purpose type=entertain degree=low>
```

Part: auxiliary tag set for corpora and collections
Member of classes:
DTD file: teicorp2
Data description: Usually empty, unless some further clarification of the type attribute is needed, in which case it may contain running prose

Occurs within: textDesc
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT purpose       - O  (%phrase.seq)                          >
<!ATTLIST purpose            %a.global
          type               CDATA                 #IMPLIED
          degree             (high | medium | low | unknown)
                                                   #IMPLIED       >
```

Discussed in: 23.2.1 ('The Text Description') on p. 541

pVar

<pVar> (i.e. pronunciation-variant reference) in a dictionary example, indicates a reference to variant pronunciation(s) of the headword.

Attributes: [None: global and inherited attributes only.]
Part: base tag set for printed dictionaries
Member of classes: dictionaries, formPointers [and indirectly also:] phrase
DTD file: teidict2
Data description: Character data or phrase-level elements.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA pRef
Declaration:

```
<!ELEMENT pVar          - -  (#PCDATA | pRef)*                      >
<!ATTLIST pVar               %a.global
                             %a.dictionaries
                             %a.formPointers                       >
```

Discussed in: 12 ('Print Dictionaries') on p. 269

q

<q> (i.e. quoted speech or thought) contains a quotation or apparent quotation — a representation of speech or thought marked as being quoted from someone else (whether in fact quoted or not); in narrative, the words are usually those of of a character or speaker; in dictionaries, <q> may be used to mark real or contrived examples of usage.

Attributes:

type may be used to indicate whether the quoted matter is spoken or thought, or to characterize it more finely.
Data type: CDATA
Sample values include:
spoken representation of direct speech, usually marked by quotation marks.
thought representation of thought, e.g. internal monologue.
Default: #IMPLIED

direct may be used to indicate whether the quoted matter is regarded as direct or indirect speech.
Data type: (Y | N | UNSPECIFIED)
Sample values include:

y speech or thought is represented directly.
n speech or thought is represented indirectly, e.g. by use of a marked verbal aspect.
unspecified no claim is made.

Default: UNSPECIFIED

who identifies the speaker of a piece of direct speech.

Data type: CDATA
Value: may be an idref
Default: #IMPLIED

Example:

```
And so, as Tiny Tim observed,
<q type=speech>God Bless Us, Every One!</q>
```

Part: base tag set for common core features
Member of classes: hqinter [and indirectly also:] common, inter
DTD file: teicore2
Occurs within: add admin argument body camera caption case castList cell cit colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition eg emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph entry entryFree event expan eTree figure foreign formula gap gi gloss graph handShift hi kinesic l label lang lg lg1 link list listBibl m measure mentioned move name note num orig oRef oVar p pause phr ptr pRef pVar q quote ref reg rs s seg shift sic sound soCalled sp stage superentry table tag tech term termEntry text time timeRange timeStruct title tree u val view vocal w writing xptr xref TEI...end
Declaration:

```
<!ELEMENT q             - -  (%specialPara)                    >

<!ATTLIST q                  %a.global
          type               CDATA               #IMPLIED
          direct             (y | n | unspecified)
                                                 unspecified
          who                CDATA               #IMPLIED       >
```

Discussed in: 6.3.3 ('Quotation') on p. 127

<quotation> specifies editorial practice adopted with respect to quotation marks in the original. quotation

Attributes:

marks indicates whether or not quotation marks have been retained as content within the text.

Data type: (NONE | SOME | ALL)
Sample values include:

none no quotation marks have been retained
some some quotation marks have been retained
all all quotation marks have been retained

Default: ALL

form specifies how quotation marks are indicated within the text.

Data type: (DATA | REND | STD | NONSTD | UNKNOWN)
Sample values include:

data quotation marks are retained as data.
rend the rendition attribute is consistently used to indicate the form of quotation marks.
std use of quotation marks has been standardized.
nonstd quotation marks are represented inconsistently.
unknown use of quotation marks is unknown.

Default: UNKNOWN

Example:

```
<quotation marks=none form=rend>
  <p>No quote marks have been retained. Instead, the rendition
  attribute on the Q element is used to specify what kinds of
  quote mark was used, according to the following list:
  <list type=gloss>
    <label>dq  <item>double quotes open and close
    <label>sq  <item>single quotes open and close
    <label>dash<item>long dash open
    <label>dg  <item>double guillemets open and close
  </list>
</quotation>
```

Example:

```
<quotation marks=all form=std>
<p>All quotation marks are retained in the text and are
 represented by standard entity references
</quotation>
```

Part: auxiliary tag set for TEI headers
Member of classes: declarable
DTD file: teihdr2
Data description:
Occurs within: editorialDecl
May contain: p
Declaration:

```
<!ELEMENT quotation     - O  (p+)                              >

<!ATTLIST quotation           %a.global
                              %a.declarable
          marks               (none | some | all) all
          form                (data | rend | std | nonstd |
                              unknown)            unknown      >
```

Discussed in: 5.3.3 ('The Editorial Practices Declaration') on p. 96; 23.3.2 ('Declarable Elements') on p. 552

quote

<quote> (i.e. quotation) contains a phrase or passage attributed by the narrator or author to some agency external to the text.

Attributes: [None: global and inherited attributes only.]
Example:

```
Lexicography has shown little sign of being affected by the
work of followers of J.R. Firth, probably best summarized
in his slogan, <quote>You shall know a word by the company it
keeps</quote><ref target=FI57>(Firth, 1957)</ref>
```

If a bibliographic citation is supplied for the source of a quotation, the two may be grouped using the <cit> element.
Part: base tag set for common core features
Member of classes: hqinter [and indirectly also:] common, inter
DTD file: teicore2
Occurs within: add admin argument body camera caption case castList cell cit colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition eg emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph entry entryFree event expan eTree figure foreign formula gap

gi gloss graph handShift hi kinesic l label lang lg lg1 link list listBibl m measure mentioned move name note num orig oRef oVar p pause phr ptr pRef pVar q quote ref reg rs s seg shift sic sound soCalled sp stage superentry table tag tech term termEntry text time timeRange timeStruct title tree u val view vocal w writing xptr xref TEI...end

Declaration:

```
<!ELEMENT quote          - -  (%specialPara;)                    >
<!ATTLIST quote                %a.global                         >
```

Discussed in: 6.3.3 ('Quotation') on p. 127; 7.3 ('Groups of Texts') on p. 195

<rate> (i.e. rate value) provides a rate value or range of values for a feature. rate

Attributes:

value provides a numeric value.
Data type: CDATA
Value: A real number or integer.
Default: #REQUIRED

valueTo together with value attribute, provides a numeric range of values.
Data type: CDATA
Value: A real number or integer.
Default: #IMPLIED

unit provides a unit for a rate feature, one of a finite list that may be specified in a feature declaration.
Data type: CDATA
Value: A string, e.g. *meter*.
Default: #IMPLIED

per provides an interval for a rate feature, one of a finite list that may be specified in a feature declaration.
Data type: CDATA
Value: A string, e.g. *second*.
Default: #REQUIRED

rel indicates the relation of the given value or range to the actual value or range.
Data type: (EQ | NE | GT | GE | LT | LE)
Sample values include:

- *eq* indicates that the actual value or range is that given.
- *ne* indicates that the actual value or range is not the value or range given by the element.
- *lt* indicates that the actual value or range is less than the given value or range.
- *le* indicates that the actual value or range is less than or equal to the given value or range.
- *gt* indicates that the actual value or range is greater than the given value or range.
- *ge* indicates that the actual value or range is greater than or equal to the given value or range.

Default: EQ

type indicates whether value is to be understood as real or integer.
Data type: (INT | REAL)
Sample values include:

- *int* specifies that value is an integer; if noninteger is given as value of value, then only integer part is used.
- *real* specifies that value is that of a real number.

Default: #IMPLIED

Example:

```
<rate rel=gt value=65 unit=mile per=hour>
```

Part: additional tag set for feature structures
Member of classes: singleVal [and indirectly also:] featureVal
DTD file: teifs2

Data description: Empty tag.
Occurs within: f fvLib if vAlt vDefault vRange
May contain: [none]
Declaration:

```
<!ELEMENT rate          - O  EMPTY                              >

<!ATTLIST rate               %a.global
          value              CDATA                #REQUIRED
          valueTo            CDATA                #IMPLIED
          unit               CDATA                #IMPLIED
          per                CDATA                #REQUIRED
          rel                (eq | ne | gt | ge | lt | le)
                                                  eq
          type               (int | real)         #IMPLIED    >
```

Discussed in: 16.4 ('Symbolic, Numeric, Measurement, Rate and String Values') on p. 402

rdg

<rdg> (i.e. reading) contains a single reading within a textual variation.

Attributes: [None: global and inherited attributes only.]
Example:

```
<rdg wit='Ra2'>Eryment</rdg>
```

Part: additional tag set for text criticism
Member of classes: readings
DTD file: teitc2
Data description: May contain character data and phrase-level elements.
Occurs within: app rdgGrp
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref [May include at any level: %m.fragmentary]
Declaration:

```
<!ELEMENT rdg           - O  (%paraContent)      +(%m.fragmentary)
                                                                >

<!ATTLIST rdg                %a.global
                             %a.readings                        >
```

Discussed in: 19.1 ('The Apparatus Entry, Readings, and Witnesses') on p. 469

rdgGrp

<rdgGrp> (i.e. reading group) within a textual variation, groups two or more readings perceived to have a genetic relationship or other affinity.

Attributes: [None: global and inherited attributes only.]
Part: additional tag set for text criticism
Member of classes: readings
DTD file: teitc2
Data description: May contain readings and nested reading groups.
Occurs within: app rdgGrp
May contain: rdg rdgGrp wit
Declaration:

```
<!ELEMENT rdgGrp        - O  (rdgGrp | (rdg, wit?))+            >

<!ATTLIST rdgGrp             %a.global
                             %a.readings                        >
```

Discussed in: 19.1 ('The Apparatus Entry, Readings, and Witnesses') on p. 469

re

<re> (i.e. related entry) contains a dictionary entry for a lexical item related to the headword, such as a compound phrase or derived form, embedded inside a larger entry.

Attributes:

type classifies the related entry according to any convenient typology.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED

The following example from *Webster's New Collegiate Dictionary* (Springfield, Mass.: G. & C. Merriam Company, 1975) shows a single related entry for which no definition is given, since its meaning is held to be readily derivable from the root entry.

Example:

```
<entry>
<form>
   <orth>neu &middot; ral</orth>
   <pron>&STRESS;n(y)&udot;r-&schwa;l</pron>
   </form>
<gramGrp><pos>adj</pos></gramGrp>
<sense n='1'><def>of, relating to, or affecting a nerve or the
        nervous system</def></sense>
<sense n='2'> ... </sense>
<re>
<form>
   <orth>neurally</orth>
   <pron extent=suffix>-&schwa;-l&emacr;</pron>
   </form>
<gramGrp><pos>adv</pos></gramGrp>
</re>
</entry>
```

The following example from *Diccionario de la Universidad de Chicago Inglés-Español y Español-Inglés / The University of Chicago Spanish Dictionary*, Fourth Edition, compiled by Carlos Castillo and Otto F. Bond (Chicago: University of Chicago Press, 1987) shows a number of related entries embedded in the main entry. The original entry resembles the following: "**abeja** [a · \é · xa] *f.* bee; **abejera** [a · \e · xé · ra] *f.* beehive; **abejón** [a · \e · xóon] *m.* drone; bumblebee; **abejorro** [a · \e · xó · rro] *m.* bumble bee." One encoding for this entry would be:

Example:

```
<entry>
  <form> <orth> abeja </orth>
      <pron> [a &middot; &bslash;&eacute; &middot; xa] </pron>
      </form>
  <gramGrp> <pos> f. </pos> </gramGrp>
  <sense> <def> bee </def>; </sense>
<re>
  <form> <orth> abejera </orth>
      <pron> [a &middot; &bslash;e &middot; x&eacute; &middot; ra] </pron>
      </form>
  <gramGrp> <pos> f. </pos> </gramGrp>
  <sense> <def> beehive </def>; </sense> </re>
<re>
  <form> <orth> abej&oacute;n </orth>
      <pron> [a &middot; &bslash;e &middot; x&oacute;n] </pron>
      </form>
  <gramGrp> <pos> m. </pos> </gramGrp>
  <sense> <def> drone </def>; </sense>
  <sense> <def> bumblebee </def>; </sense> </re>
<re>
  <form> <orth> abejorro </orth>
      <pron> [a &middot; &bslash;e &middot; x&oacute; &middot; rro] </pron>
      </form>
  <gramGrp> <pos> m. </pos> </gramGrp>
  <sense> <def> bumble bee </def>. </sense> </re>
</entry>
```

In the much larger Simon & Schuster Spanish-English dictionary,[2] these derived forms of 'abeja' are treated as separate main entries, but there are other embedded phrases shown as <re>s in its main entry for 'abeja': "abeja, f. 1. (ento.) bee. 2. busy bee, hard worker. 3. (astron.) A., Musca. – a. albanila, mason bee; a. carpintera, carpenter bee; a. reina or maestra, queen bee; a. neutra or obrera, worker bee." This entry may be encoded thus:

Example:

```
<entry>
<form> <orth> abeja </orth> <gramGrp><gen> f. </gen></gramGrp> </form>
<sense n='1.'>
        <usg type=domain> (ento.) </usg>
        <def> bee </def>.
        </sense>
<sense n='2.'>
        <def> busy bee, hard worker </def>.
        </sense>
<sense n='3.'>
        <usg type=domain orig='A.'> (astron.) </usg>,
        <def> Musca </def> ---
        </sense>
<re>
<form>  <orth orig='a. albanila'> abeja albanila </orth>,
        </form>
<sense><def> mason bee </def>; </sense>
</re>
<re>
<form>  <orth orig='a. carpintera'>  abeja carpintera </orth>,
        </form>
<sense> <def> carpenter bee </def>; </sense>
</re>
<re>
<form>  <orth id=O1 orig='a. reina or maestra'> abeja reina </orth>
        <orth mergedin=O1> abeja maestra </orth>
        </form>
<sense> <def> queen bee </def>; </sense>
</re>
<re>
<form>  <orth id=O2 orig='a. neutra or obrera'> abeja neutra </orth>
        <orth mergedin=O2> abeja obrera </orth>
        </form>
<sense> <def> worker bee </def> . </sense>
</re>
</entry>
```

Identical in sub-elements to an <entry> tag, and used where a dictionary has embedded information inside one entry which could have formed a separate entry. Some authorities distinguish related entries, run-on entries, and various other types of degenerate entries; no such typology is attempted here.

Part: base tag set for printed dictionaries

Member of classes: dictionaries, dictionaryParts, dictionaryTopLevel

DTD file: teidict2

Data description: May contain character data mixed with any other elements defined in the dictionary tag set.

Occurs within: entry hom re sense trans [May not appear at any level within: re]

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct def del distinct eg emph etym expan foreign form formula gap gi gloss gramGrp handShift hi lang link m measure mentioned name note num orig oRef oVar phr ptr pRef pVar re ref reg rs s seg sense sic soCalled tag term time timeRange timeStruct title trans usg val w xptr xr xref [May not include at any level: re]

Declaration:

[2] Tana de Gámez, ed., *Simon and Schuster's International Dictionary* (New York: Simon and Schuster, 1973).

```
<!ELEMENT re            - O  (sense | %m.dictionaryTopLevel |
                              %m.phrase | #PCDATA)*
                                                  -(re)        >
<!ATTLIST re                 %a.global
                             %a.dictionaries
          type               CDATA               #IMPLIED     >
```

Discussed in: 12.3.6 ('Related Entries') on p. 296

<recording> (i.e. recording event) details of an audio or video recording event used as the source of a spoken text, either directly or from a public broadcast. recording

Attributes:

type the kind of recording.
Data type: (AUDIO | VIDEO)
Sample values include:
audio audio recording
video audio and video recording
Default: AUDIO

dur the original duration of the recording.
Data type: CDATA
Value: Include the units, e.g. 30 mins.
Default: #IMPLIED

Example:

```
<recording type=audio dur="30 mins">
<equipment>Recorded on a Sony TR444 walkman by
         unknown participants; remastered to digital tape
   at <place>Borehamwood Studios</place> by
   <name>Transcription Services</name>
```

Example:

```
<recording type=audio dur="10 mins">
  <equipment><p>Recorded from FM Radio to digital tape</>
  <broadcast>
   <bibl><title>Interview on foreign policy
            <author>BBC Radio 5
     <respStmt><resp>interviewer</><name>Robin Day</></>
     <respStmt><resp>interviewee</><name>Margaret Thatcher</></>
     <series><title>The World Tonight</></>
     <note>First broadcast on <date>27 Nov 1989</></>
 </bibl></broadcast> </recording>
```

Part: auxiliary tag set for TEI headers
Member of classes: declarable
DTD file: teihdr2
Data description:
Occurs within: broadcast recordingStmt
May contain: broadcast date equipment p respStmt
Declaration:

```
<!ELEMENT recording     - -  (p+ | (respStmt | equipment |
                              broadcast | date)*)              >
<!ATTLIST recording          %a.global
                             %a.declarable
          type               (audio | video)     audio
          dur                CDATA               #IMPLIED     >
```

Discussed in: 5.2.9 ('Computer Files Composed of Transcribed Speech ') on p. 91; 23.3.2 ('Declarable Elements') on p. 552

<recordingStmt> (i.e. recording statement) describes a set of recordings used in transcription of a spoken text. recordingStmt

Attributes: [None: global and inherited attributes only.]
Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:
Occurs within: sourceDesc
May contain: p recording
Declaration:

```
<!ELEMENT recordingStmt - -  (p+ | recording+ )                >
<!ATTLIST recordingStmt      %a.global                         >
```

Discussed in: 5.2.9 ('Computer Files Composed of Transcribed Speech ') on p. 91; 5.2.7 ('The Source Description') on p. 90

ref

<ref> defines a reference to another location in the current document, in terms of one or more identifiable elements, possibly modified by additional text or comment.

Attributes:

target specifies the destination of the reference as one or more SGML identifiers
Data type: IDREFS
Value: Each value specified must be the same as that specified as value for an ID attribute for some other element in the current SGML document.
Default: #IMPLIED

Example:

```
<ref target="P2 P5" resp=LB type=s>
   See especially the second sentence</ref>.

   See also <ref>s.v. <term>locution</term></ref>.
```

Part: base tag set for common core features
Member of classes: loc, pointer, terminologyInclusions [and indirectly also:] phrase
DTD file: teicore2
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cit cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term termEntry time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT ref           - -  (%paraContent)                   >
<!ATTLIST ref                %a.global
                             %a.pointer
          target             IDREFS              #IMPLIED    >
```

Discussed in: 6.6 ('Simple Links and Cross References') on p. 147; 13.2 ('Tags for Terminological Data') on p. 312

refsDecl

<refsDecl> (i.e. references declaration) specifies how canonical references are constructed for this text.

Attributes:

doctype identifies the *document type* within which this reference declaration is used.
Data type: NAME
Value: must be the name of a document type
Default: TEI.2

Example:

```
<refsDecl>
  <step gi=div1 att=n delim=":"
        from="DESCENDANT (1 DIV1 N %1) CHILD (1 HEAD) STRING %2"
        to  ="PARENT"
        refunit="chapter" >
  <step length=4>
        from="CHILD (ALL DIV2 N %3)"
        refunit="section" >
</refsDecl>
```

This example is a formal representation for the referencing scheme described informally in the following example:

Example:

```
<refsDecl>
  <p>References are made up by concatenating the value for
  the <att>n</att> attribute on each <gi>div1</gi> element,
  followed by a colon, followed by the first six characters
  of the first <gi>head</gi> element found within it and a
  space, followed by exactly four characters derived from the
  value of the <att>n</att> attribute on the next included
  <gi>div2</gi> element.
</refsDecl>
```

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:
Occurs within: encodingDesc
May contain: p state step
Declaration:

```
<!ELEMENT refsDecl      - O  (p+ | step+ | state+)              >
<!ATTLIST refsDecl           %a.global
          doctype            NAME                TEI.2        >
```

Discussed in: 5.3.5.3 ('Milestone Method') on p. 102; 5.3 ('The Encoding Description') on p. 93; 5.3.5 ('The Reference System Declaration ') on p. 100

<reg> (i.e. regularization) contains a reading which has been regularized or normalized in some sense. reg

Attributes:

orig (i.e. original) gives the unregularized form of the text as found in the source copy.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED
Example:

```
<reg orig='auctoritee'>Authority</reg>
```

resp (i.e. responsibility) identifies the individual responsible for the regularization of the word or phrase.
Data type: CDATA
Value: any string of characters, typically the initials of the individual involved, or a role identifier like "editor" if not known by name.
Default: #IMPLIED

The <reg> tag is mirrored by the <orig> tag, which allows the unnormalized form of the original to be retained as the content of the element, while still providing the opportunity to record the regularized or normalized form postulated by a researcher. The choice between the two elements is up to the encoder.

Part: additional tag set for common core features
Member of classes: edit [and indirectly also:] phrase
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT reg           - -  (%phrase.seq;)                    >

<!ATTLIST reg                %a.global
          orig               CDATA                 #IMPLIED
          resp               CDATA                 #IMPLIED    >
```

Discussed in: 6.5.2 ('Regularization and Normalization') on p. 143; 19 ('Critical Apparatus') on p. 467

region

<region> (i.e. region) in an address, contains the state, province, county or region name; in a place name given as a hierarchy of geo-political units, the <region> is larger or administratively superior to the <settlement> and smaller or administratively less important than the <country>.

Attributes: [None: global and inherited attributes only.]
Example:

```
<placeName>
  <region type=state n='IL'>Illinois
</placeName>
```

Part: additional tag set for
Member of classes: placePart [and indirectly also:] names
DTD file: teind2
Occurs within: placeName
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT region        - -  (%paraContent)                    >

<!ATTLIST region             %a.global
                             %a.placePart                      >
```

Discussed in: 20.2 ('Place Names') on p. 493

<relation> (i.e. relationship) describes any kind of relationship or linkage amongst a specified group of participants. relation

Attributes:

type categorizes the relationship in some respect, e.g. as social, personal or other.
Data type: CDATA
Sample values include:
social relationship concerned with social roles
personal relationship concerned with personal roles, e.g. kinship, marriage, etc.
other other kinds of relationship
Default: PERSONAL

desc briefly describes the relationship.
Data type: CDATA
Value: an open list of application-dependent keywords
Default: #IMPLIED

active identifies the "active"participants in a non-mutual relationship, or all the participants in a mutual one.
Data type: IDREFS
Value: a list of identifier values for participant or participant groups
Default: #IMPLIED

passive identifies the "passive" participants in a non-mutual relationship.
Data type: IDREFS
Value: a list of identifier values for participant or participant groups
Default: #IMPLIED

mutual indicates whether the relationship holds equally amongst all the participants.
Data type: (Y | N)
Sample values include:
Y the relationship is mutual
N the relationship is directed
Default: Y

Example:

```
<relation type=social desc="supervisor"
     active=P1 passive="P2 P3 P4" mutual=N>
```

Example:

```
<relation type=personal desc="friends"
     active="P2 P3 P4" mutual=Y>
```

Part: auxiliary tag set for corpora and collections
Member of classes:
DTD file: teicorp2
Data description: Empty
Occurs within: particLinks
May contain: [none]
Declaration:

```
<!ELEMENT relation      - O  EMPTY                              >

<!ATTLIST relation           %a.global
          type               CDATA               personal
          desc               CDATA               #IMPLIED
          active             IDREFS              #IMPLIED
          passive            IDREFS              #IMPLIED
          mutual             (Y | N)             Y               >
```

Discussed in: 23.2.2 ('The Participants Description') on p. 545

<remarks> contains any commentary or discussion about the usage of an element, attribute, class, or entity not otherwise documented within the containing element. remarks

Attributes: [None: global and inherited attributes only.]
Example:

```
<remarks><p>This element is probably redundant.
```

As defined in ODD, must contain paragraphs; should be special.para
Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: Contains at least one paragraph, unless it is empty.
Occurs within: attDef classDoc entDoc tagDoc
May contain: bibl biblFull biblStruct camera caption castList cit entry entryFree event eTree graph kinesic l label lg lg1 list listBibl move note p pause q quote shift sound sp stage superentry tech termEntry tree u view vocal writing TEI...end
Declaration:

```
<!ELEMENT remarks       - O  (%component.seq)                  >
<!ATTLIST remarks            %a.global                         >
```

Discussed in: 27.1 ('The TagDoc Documentation Element') on p. 603; 27.1.1 ('The AttList Documentation Element') on p. 605; 27.2 ('Element Classes') on p. 606; 27.3 ('Entity Documentation ') on p. 607

rendition

<rendition> (i.e. rendition) supplies information about the intended rendition of one or more elements.
Attributes: [None: global and inherited attributes only.]
Example:

Future releases of these Guidelines will provide for more detailed specifications of rendition in terms of DSSSL properties; the present proposal provides a hook for these extensions only.
Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Occurs within: tagsDecl
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT rendition     - O  (%paraContent)                    >
<!ATTLIST rendition          %a.global                         >
```

Discussed in: 5.3.4 ('The Tagging Declaration') on p. 98

residence

<residence> (i.e. residence) describes a person's present or past places of residence.
Attributes: [None: global and inherited attributes only.]
Example:

```
<residence>Childhood in East Africa, long term
  resident of Glasgow, Scotland.
```

Part: auxiliary tag set for corpora and collections
Member of classes: demographic
DTD file: teicorp2
Data description: May contain character data and phrase-level elements.
Occurs within: person personGrp
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT residence     - O  (%phrase.seq)                    >
<!ATTLIST residence          %a.global                        >
```

Discussed in: 23.2.2 ('The Participants Description') on p. 545

<resp> contains a phrase describing the nature of a person's intellectual responsibility. resp

Attributes: [None: global and inherited attributes only.]

Example:

```
<respStmt><resp>compiler</resp> <name>Edward Child</name></respStmt>
```

Part: auxiliary tag set for TEI headers

Member of classes:

DTD file: teicore2

Data description:

Occurs within: respStmt

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT resp          - O  (%phrase.seq;)                   >
<!ATTLIST resp               %a.global                        >
```

Discussed in: 6.10.2.2 ('Authors, Titles, and Editors') on p. 168; 5.2.1 ('The Title Statement') on p. 82; 5.2.2 ('The Edition Statement') on p. 84; 5.2.5 ('The Series Statement') on p. 88

<respons> (i.e. responsibility) identifies the individual(s) responsible for some aspect of the markup of some particular element(s). respons

Attributes:

target gives the SGML identifier(s) of the element(s) for which some aspect of the responsibility is being assigned.
Data type: IDREFS
Value: one or more SGML identifiers.
Default: #REQUIRED

locus indicates the specific aspect of the markup for which responsibility is being assigned.
Data type: CDATA
Sample values include:

#gi responsibility for the claim that the element is of the type indicated by the markup

#location responsibility for the claim that the element begins and ends where indicated

#startloc responsibility for the claim that the element begins where indicated

#endloc responsibility for the claim that the element ends where indicated

name responsibility for the claim that the name attribute has the value given in the markup

#transcribedContent responsibility for the transcription of the element content

#suppliedContent responsibility for the contents supplied by the encoder (corrections, expansions of abbreviations, etc.)

Default: #REQUIRED

resp identifies the individual or agency responsible for the indicated aspect of the electronic text.
Data type: CDATA
Value: any string of characters, typically the initials of an individual, the acronym of an agency, the name of a computer program, etc.
Default: #REQUIRED

desc (i.e. description) gives a brief prose note supplying any additional information which should be recorded

Data type: CDATA
Value: any string of characters, typically a phrase or sentence in a natural language.
Default: #IMPLIED

Example:

```
<respons target=p1
         locus='#gi #location' resp=AR>
 <respons target=p2
         locus=rend resp=LB>
```

The <respons> element is designed for cases in which fine-grained information about specific aspects of the markup of a text is desirable for whatever reason. Global responsibility for certain aspects of markup is usually more simply indicated in the TEI header, using the <respStmt> element within the title statement, edition statement, or change log.
Part: additional tag set for
Member of classes: metadata [and indirectly also:] globincl
DTD file: teicert2
Data description: Empty.
Occurs within: [none]
May contain: [none]
Declaration:

```
<!ELEMENT respons       - O  EMPTY                          >

<!ATTLIST respons            %a.global
          target             IDREFS             #REQUIRED
          locus              CDATA              #REQUIRED
          resp               CDATA              #REQUIRED
          desc               CDATA              #IMPLIED    >
```

Discussed in: 17.2 ('Attribution of Responsibility') on p. 440

respStmt

<respStmt> (i.e. statement of responsibility) supplies a statement of responsibility for someone responsible for the intellectual content of a text, edition, recording, or series, where the specialized elements for authors, editors, etc. do not suffice or do not apply.

Attributes: [None: global and inherited attributes only.]
Example:

```
<respStmt><resp>transcribed from original ms</resp>
      <name>Claus Huitfeldt</name>
</respStmt>
```

Example:

```
<respStmt><resp>converted to SGML encoding</resp>
      <name>Alan Morrison</name>
</respStmt>
```

Part: base tag set for common core features
Member of classes: biblPart
DTD file: teicore2
Data description:
Occurs within: analytic bibl change editionStmt monogr recording series seriesStmt titleStmt
May contain: name resp
Declaration:

```
<!ELEMENT respStmt      - O  ((resp & name), (resp | name)*)    >

<!ATTLIST respStmt           %a.global                          >
```

Discussed in: 6.10.2.2 ('Authors, Titles, and Editors') on p. 168; 5.2.1 ('The Title Statement') on p. 82; 5.2.2 ('The Edition Statement') on p. 84; 5.2.5 ('The Series Statement') on p. 88

restore

<restore> indicates restoration of text to an earlier state by cancellation of an editorial or authorial marking or instruction.

Attributes:

type indicates the action cancelled by the restoration.
Data type: CDATA
Value: The value of this should be the name of the tag contained within the <restore> element which is cancelled by the restoration. Most often, this will be <del>, but might also be <hi>, etc. In cases of simple nesting of a single cancelled action within the <restore> element this attribute will not be necessary
Default: #IMPLIED

desc (i.e. description) gives a prose description of the means of restoration.
Data type: CDATA
Value: Any word or phrase, such as "stet" or "strike-down".
Default: #IMPLIED

resp (i.e. responsible) signifies the editor or transcriber responsible for identifying the hand of the restoration.
Data type: IDREF
Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).
Default: %INHERITED

cert (i.e. certainty) signifies the degree of certainty ascribed to the identification of the hand of the restoration.
Data type: CDATA
Default: #IMPLIED

hand signifies the hand of the agent which made the restoration.
Data type: IDREF
Value: must be one of the hand identifiers declared in the document header (see section 18.2.1 ('Document Hands') on p. 456).
Default: %INHERITED

Remainder of this tagdoc copied from rdg by LB

Part: additional tag set for text criticism
Member of classes: readings
DTD file: teitran2
Data description: May contain character data and phrase-level elements.
Occurs within: [none]
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT restore       - O  (%phrase.seq;)                    >

<!ATTLIST restore            %a.global
          wit                CDATA               #IMPLIED
          cause              CDATA               #IMPLIED
          varSeq             NUMBER              #IMPLIED
          type               CDATA               #IMPLIED
          desc               CDATA               #IMPLIED
          resp               IDREF               %INHERITED

          cert               CDATA               #IMPLIED
          hand               IDREF               %INHERITED     >
```

Discussed in: 18.1.6 ('Cancellation of Deletions and Other Markings') on p. 454

<revisionDesc> (i.e. revision description) summarizes the revision history for a file. revisionDesc

Attributes: [None: global and inherited attributes only.]
Example:

```
<revisionDesc>
<change><date>11 Nov 91  <name>EB  <what>Deleted chapter 10
```

```
<change><date>23 Sept 91 <name>MSM <what>First draft
</revisionDesc>
```

Record changes with most recent changes at the top of the list.
Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:
Occurs within: teiHeader
May contain: change list
Declaration:

```
<!ELEMENT revisionDesc  - -  (list | change+)                >
<!ATTLIST revisionDesc       %a.global                       >
```

Discussed in: 5.5 ('The Revision Description') on p. 112; 5.1.1 ('The TEI Header and Its Components') on p. 78

role

<role> the name of a dramatic role, as given in a cast list.

Attributes: [None: global and inherited attributes only.]
Example:

```
<role id=JT>Joan Trash</><roleDesc>A Ginger-bread-woman</>
```

It is important to assign a meaningful ID attribute to the <role> element, since this ID is referred to by who attributes on many other elements.
Part: base tag set for performance texts
Member of classes:
DTD file: teidram2
Data description:
Occurs within: castItem
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT role          - O  (%phrase.seq)                   >
<!ATTLIST role               %a.global                       >
```

Discussed in: 10.1.4 ('Cast Lists') on p. 233

roleDesc

<roleDesc> (i.e. role description) describes a character's role in a drama.

Attributes: [None: global and inherited attributes only.]
Example:

```
<roleDesc>gentlemen of leisure
```

Part: base tag set for performance texts
Member of classes:
DTD file: teidram2
Data description:
Occurs within: castItem
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT roleDesc      - -  (%phrase.seq)                   >
<!ATTLIST roleDesc           %a.global                       >
```

Discussed in: 10.1.4 ('Cast Lists') on p. 233

roleName

<roleName> (i.e. roleName) contains a name component which indicates that the referent has a particular role or position in society, such as an official title or rank.

Attributes: [None: global and inherited attributes only.]
Example:

```
<persName>
  <foreName>William</foreName>
  <surname>Poulteny</surname>
  <roleName>Earl of Bath</roleName>
</persName>
```

A <roleName> may be distinguished from an <addName> by virtue of the fact that, like a title, it typically exists independently of its holder.
Part: additional tag set for names and dates
Member of classes: personPart [and indirectly also:] names
DTD file: teind2
Occurs within: persName
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT roleName      - -  (%phrase.seq)                    >

<!ATTLIST roleName           %a.global
                             %a.personPart                    >
```

Discussed in: 20.1 ('Personal Names') on p. 488

<root> (i.e. root node) represents the root node of a tree. root

Attributes:

label gives a label for a root node.
Data type: CDATA
Value: A character string.
Default: #IMPLIED

value provides the value of the root, which is a feature structure or other analytic element.
Data type: IDREF
Value: A valid identifier of a feature structure or other analytic element.
Default: #IMPLIED

children provides a list of IDs of the elements which are the children of the root node.
Data type: IDREFS
Value: A list of valid identifiers.
Default: #IMPLIED
If the root has no children (i.e., the tree is "trivial"), then the children attribute must be omitted. For technical reasons, it cannot be specified as <root children="">.

ord indicates whether or not the root is ordered.
Data type: (Y | N)
Sample values include:

Y indicates that the children of the root are ordered.

N indicates that the children of the root are unordered.

Default: #IMPLIED
Use if and only if ord=partial is specified on the <tree> tag and the root has more than one child.

outDegree gives the out degree of the root, the number of its children.
Data type: NUMBER
Value: A nonnegative integer.
Default: #IMPLIED
The in degree of the root is always 0.

Example:

```
<root id=VP1 label=VP children='VB1 PN1' outDegree=2>
```

Part: additional tag set for graphs, networks, and trees
Member of classes:
DTD file: teinet2
Data description: empty
Occurs within: tree
May contain: [none]
Declaration:

```
<!ELEMENT root          - O  EMPTY                          >

<!ATTLIST root               %a.global
          label              CDATA              #IMPLIED
          value              IDREF              #IMPLIED
          children           IDREFS             #IMPLIED
          ord                (Y | N)            #IMPLIED
          outDegree          NUMBER             #IMPLIED    >
```

Discussed in: 21.2 ('Trees') on p. 514

row

<row> contains one row of a table.

Attributes:

role indicates the kind of information held in the cells of this row.
Data type: CDATA
Sample values include:
label labelling or descriptive information only.
data data values.
Default: DATA
The value specified is the default for all cells in this row.

Example:

```
<row role=data>
  <cell role=label>Classics
  <cell>Idle listless and unimproving
</row>
```

Part: base tag set for formulae
Member of classes:
DTD file: teifig2
Data description:
Occurs within: table
May contain: cell table
Declaration:

```
<!ELEMENT row           - O  ((cell | table)+)              >

<!ATTLIST row                %a.global
          role               CDATA              data     >
```

Discussed in: 22.1.1 ('The TEI Table DTD') on p. 524

rs

<rs> (i.e. referencing string) contains a general purpose name or referring string.

Attributes:

type indicates more specifically the object referred to by the referencing string. Values might include "person", "place", "ship", "element" etc.
Data type: CDATA
Value: Any string of characters.
Default: #IMPLIED

Example:

```
<q>My dear <rs type=person>Mr. Bennet</rs>, </q>
said <rs type=person>his lady</rs> to him one day,
<q>have you heard that <rs type=place>Netherfield
Park</rs> is let at last?</q>
```

Part: common tag set for common core features
Member of classes: data, names [and indirectly also:] phrase
DTD file: teicore2
Occurs within: abbr activity actor add addrLine addName admin affiliation attDef author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode classDoc closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph entDoc equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagDoc tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT rs            - -  (%phrase.seq)                    >
<!ATTLIST rs                 %a.global
                             %a.names
          type               CDATA                #IMPLIED      >
```

Discussed in: 20.1 (‘Personal Names’) on p. 488; 6.4.1 (‘Referring Strings’) on p. 132

<s> (i.e. s-unit) contains a sentence-like division of a text. S

Attributes: [None: global and inherited attributes only.]
Example:

```
<s>When are you leaving?</s>
<s>Tomorrow.</s>
```

The <s> element may be used to mark orthographic sentences, or any other segmentation of a text, provided that the segmentation is end-to-end, complete, and non-nesting. For other kinds of segmentation, the <seg> element should be used.
Part: additional tag set for common core features
Member of classes: seg [and indirectly also:] phrase
DTD file: teiana2
Data description: May contain character data, phrase-level and segmentation class elements, other than <s>.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref [May not appear at any level within: s]
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref [May not include at any level: s]
Declaration:

```
<!ELEMENT s              - -  (%phrase.seq)        -(s)          >
<!ATTLIST s                   %a.global
                              %a.seg                             >
```

Discussed in: 15.1 ('Linguistic Segment Categories') on p. 382; 11.3.1 ('Segmentation') on p. 261

salute

<salute> (i.e. salutation) contains a salutation or greeting prefixed to a foreword, dedicatory epistle or other division of a text, or the salutation in the closing of a letter, preface, etc.

Attributes: [None: global and inherited attributes only.]
Example:

```
<salute>To all courteous mindes, that will voutchsafe the
readinge.</>
```

Part: base tag set for common core features
Member of classes: divbot, divtop
DTD file: teistr2
Occurs within: back body castList closer div div0 div1 div2 div3 div4 div5 div6 div7 epilogue front group lg opener performance prologue
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT salute         - O  (%phrase.seq;)                    >
<!ATTLIST salute              %a.global                         >
```

Discussed in: 7.2.4 ('Content of Textual Divisions') on p. 194; 7.2.2 ('Openers and Closers') on p. 192

samplingDecl

<samplingDecl> (i.e. sampling declaration) contains a prose description of the rationale and methods used in sampling texts in the creation of a corpus or collection.

Attributes: [None: global and inherited attributes only.]
This element records all information about systematic inclusion or omission of portions of the text, whether a reflection of sampling procedures in the pure sense or of systematic omission of material deemed either too difficult to transcribe or not of sufficient interest.
Part: auxiliary tag set for TEI headers
Member of classes: declarable
DTD file: teihdr2
Data description:
Occurs within: encodingDesc
May contain: p
Declaration:

```
<!ELEMENT samplingDecl   - O  (p+)                              >
<!ATTLIST samplingDecl        %a.global
                              %a.declarable                     >
```

Discussed in: 5.3.2 ('The Sampling Declaration') on p. 95; 5.3 ('The Encoding Description') on p. 93; 23.3.2 ('Declarable Elements') on p. 552

script

<script> contains a prose description of the script declared by a writing system declaration.

Attributes: [None: global and inherited attributes only.]
Example:

```
<writingSystemDeclaration lang=eng id='iso8859-1'
    name='-//TEI P2: 1993//WSD ISO 8859-1: 1993//en'
    date='1993-06-01'>
 <language iso639=''>Various</language>
 <script>Latin script with diacritics.</script>
 <!-- ... -->
</writingSystemDeclaration>
```

Example:

```
<writingSystemDeclaration lang=eng id=jpn
 name='-//TEI P2: 1993//WSD JIS 0208//en' date='1993-06-01'>
 <language iso639='jpn'>Modern Japanese</language>
 <script>normal Japanese, with mixture of hiragana, katakana,
        and kanji.</script>
 <!-- ... -->
</writingSystemDeclaration>
```

This element is provided for the sake of clarity and readability. Strictly speaking, it is almost always redundant, as knowledgeable readers will know what script is involved as soon as they examine the base character set or the character repertoire.

Part: auxiliary tag set for writing system declarations
Member of classes:
DTD file: teiwsd2
Data description: May contain character data only.
Occurs within: writingSystemDeclaration
May contain: #PCDATA
Declaration:

```
<!ELEMENT script        - O  (#PCDATA)                          >
<!ATTLIST script             %a.global                          >
```

Discussed in: 25.3 ('Describing the Writing System') on p. 574

<scriptStmt> (i.e. script statement) contains a citation giving details of the script used for a spoken text. — scriptStmt

Attributes: [None: global and inherited attributes only.]
Part: auxiliary tag set for TEI headers
Member of classes: declarable
DTD file: teihdr2
Data description:
Occurs within: sourceDesc
May contain: bibl biblFull biblStruct p
Declaration:

```
<!ELEMENT scriptStmt    - -  (p+ | bibl | biblFull |
                              biblStruct)                       >
<!ATTLIST scriptStmt         %a.global
                             %a.declarable                      >
```

Discussed in: 5.2.9 ('Computer Files Composed of Transcribed Speech ') on p. 91; 5.2.7 ('The Source Description') on p. 90; 23.3.2 ('Declarable Elements') on p. 552

<second> (i.e. second) the second component of a structured time-expression. — second

Attributes: [None: global and inherited attributes only.]
Example:

```
At the third stroke the time sponsored by Accurist
will be
<timeStruct  value='1821:30'>
  <hour>six</hour>
  <minute>twenty-one</minute> and
  <second>thirty</second> seconds
</timeStruct>
```

Part: additional tag set for names and dates
Member of classes: temporalExpr
DTD file: teind2
Occurs within: dateStruct timeStruct
May contain: #PCDATA
Declaration:

```
<!ELEMENT second        - -  (#PCDATA)                    >
<!ATTLIST second             %a.global
                             %a.temporalExpr               >
```

Discussed in: 20.4 ('Dates and Time') on p. 499

seg

<seg> (i.e. arbitrary segment) contains any arbitrary phrase-level unit of text (including other <seg> elements).

Attributes:

subtype provides a sub-categorization of the segment is marked.

Data type: CDATA

Value: any string of characters.

Default: #IMPLIED

The subtype attribute may be used to provide any classification for the <seg> elements tagged in a text suitable for the type given.

part specifies whether or not the segment is complete.

Data type: (Y | N | I | M | F)

Sample values include:

Y the segment is incomplete

N either the segment is complete, or no claim is made as to its completeness

I the initial part of an incomplete segment

M a medial part of an incomplete segment

F the final part of an incomplete segment

Default: N

The values I, M, or F should be used only where it is clear how the segment is to be reconstituted.

Example:

```
<seg>When are you leaving?</seg>
<seg>Tomorrow.</seg>
```

Example:

```
<s>
<seg type='typographic part-line'
rend=caps>So father's only</seg>
glory was the ballfield.</s>
```

Example:

```
<seg id=S1S3 type=preamble>
  <seg id=S1>Sigmund,
      <seg type=patronym>the son of Volsung</seg>,
  was a king in Frankish country.</seg>
  <seg id=S2>Sinfiotli was the eldest of his sons ...</seg>
  <seg id=S3>Borghild, Sigmund's wife, had a brother ... </seg>
<!-- ... -->
</seg>
```

The <seg> element may be used at the encoder's discretion to mark any segments of the text of interest for processing. One use of the element is to mark text features for which no appropriate markup is otherwise defined — i.e. as a simple extension mechanism. Another use is to provide an identifier for some segment which is to be pointed at by some other element — i.e. to provide a target, or a part of a target, for a <ptr> or other similar element.

Part: additional tag set for common core features

Member of classes: seg [and indirectly also:] phrase

DTD file: teilink2.dtd

Data description: May contain anything which may appear within a paragraph.

Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor

docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale m measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled sp speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view w wit witness witDetail writing xr xref

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref

Declaration:

```
<!ELEMENT seg           - -  (%paraContent)                    >
<!ATTLIST seg                %a.global
                             %a.seg
          subtype            CDATA                #IMPLIED
          part               (Y | N | I | M | F) N             >
```

Discussed in: 14.3 ('Segments and Anchors') on p. 355; 9.3 ('Components of the Verse Line') on p. 218; 10.2.4 ('Speech Contents') on p. 241

segmentation

<segmentation> describes the principles according to which the text has been segmented, for example into sentences, tone-units, graphemic strata, etc.

Attributes: [None: global and inherited attributes only.]
Part: auxiliary tag set for TEI headers
Member of classes: declarable
DTD file: teihdr2
Data description:
Occurs within: editorialDecl
May contain: p
Declaration:

```
<!ELEMENT segmentation  - O  (p+)                              >
<!ATTLIST segmentation       %a.global
                             %a.declarable                     >
```

Discussed in: 5.3.3 ('The Editorial Practices Declaration') on p. 96; 23.3.2 ('Declarable Elements') on p. 552

sense

<sense> (i.e. sense information group) groups together all information relating to one word sense in a dictionary <entry> (definitions, examples, translation equivalents, etc.)

Attributes:

level gives the nesting depth of this sense.
Data type: NUMBER
Value: any string of digits
Default: #IMPLIED

Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts
DTD file: teidict2
Data description: May contain character data mixed with any other elements defined in the dictionary tag set.
Occurs within: entry hom re sense trans
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct def del distinct eg emph etym expan foreign form formula gap gi gloss gramGrp handShift hi lang link m measure mentioned name note num orig oRef oVar phr ptr pRef pVar re ref reg rs s seg sense sic soCalled tag term time timeRange timeStruct title trans usg val w xptr xr xref

Declaration:

```
<!ELEMENT sense          - -  (sense | %m.dictionaryTopLevel |
                               %m.phrase | #PCDATA)*              >

<!ATTLIST sense                %a.global
                               %a.dictionaries
          level                NUMBER              #IMPLIED      >
```

Discussed in: 12.2 ('The Structure of Dictionary Entries') on p. 274

series

<series> (i.e. series information) contains information about the series in which a book or other bibliographic item has appeared.

Attributes: [None: global and inherited attributes only.]
Part: base tag set for common core features
Member of classes: biblPart
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: bibl biblStruct
May contain: biblScope editor respStmt title
Declaration:

```
<!ELEMENT series         - O  (title | editor | respStmt |
                               biblScope)*                       >

<!ATTLIST series               %a.global                         >
```

Discussed in: 6.10.2.1 ('Analytic, Monographic, and Series Levels') on p. 166

seriesStmt

<seriesStmt> (i.e. series statement) groups information about the *series*, if any, to which a publication belongs.

Attributes: [None: global and inherited attributes only.]
Example:

```
<seriesStmt>
    <title>Machine-Readable Texts for the Study of Indian
                  Literature</title>
    <respStmt>
         <resp>ed. by</resp> <name>Jan Gonda</name>
    </respStmt>
    <idno type=vol>1.2</idno>
    <idno type=ISSN>0 345 6789</idno>
</seriesStmt>
```

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:
Occurs within: biblFull fileDesc
May contain: idno p respStmt title
Declaration:

```
<!ELEMENT seriesStmt     - O  ( (title, (idno | respStmt)*) | p+
                               )                                 >

<!ATTLIST seriesStmt           %a.global                         >
```

Discussed in: 5.2.5 ('The Series Statement') on p. 88; 5.2 ('The File Description') on p. 80

set

<set> contains a description of the setting, time, locale, appearance, etc., of the action of a play, typically found in the front matter of a printed performance text (not a stage direction).

Attributes: [None: global and inherited attributes only.]
Example:

```
<set>
<p>The action takes place on February 7th between the hours of noon
```

```
and six in the afternoon, close to the Trenartha Tin Plate Works,
on the borders of England and Wales, where a strike has been in
progress throughout the winter.
</set>
```

Example:

```
<set><head>SCENE</>
A Sub-Post Office on a late autumn evening
</set>
```

Example:

```
<front>
  <titlePage> ... </titlePage>
  <div type='Notice'> ... </>
  <div type='Dedication'> ... </>
  <div type='Performance'> ... </>
  <set>
    <list type=gloss>
      <label>TIME: <item>1907
      <label>PLACE: <item>East Coast village in England
    </list>
  </set>
</front>
```

This element should not be used outside the front matter; for similar contextual descriptions within the body of the text, use the <stage> element.

Part: base tag set for performance texts

Member of classes: dramafront [and indirectly also:] front

DTD file: teidram2

Data description: Contains paragraphs or phrase level tags.

Occurs within: back front

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph entry entryFree event expan eTree figure foreign formula gap gi gloss graph handShift head hi kinesic l label lang lg lg1 link list listBibl m measure mentioned move name note num orig oRef oVar p pause phr ptr pRef pVar q quote ref reg rs s seg shift sic sound soCalled sp stage superentry table tag tech term termEntry text time timeRange timeStruct title tree u val view vocal w writing xptr xref TEI...end

Declaration:

```
<!ELEMENT set           - -  (head?, %specialPara)             >
<!ATTLIST set                %a.global                         >
```

Discussed in: 10.1 ('Front and Back Matter ') on p. 228

setting

<setting> (i.e. setting) describes one particular setting in which a language interaction takes place.

Attributes:

who supplies the identifiers of the participants at this setting.
Data type: IDREFS
Value: must correspond with ID values of <participant> or <participant.grp> elements in the current document.
Default: #IMPLIED

Example:

```
<setting who=P1>
<name>New York City, US
<time>1989
<locale>on a park bench
<activity>feeding birds
</setting>
```

If the who attribute is not supplied, the setting is assumed to be that of all participants in the language interaction.

Part: auxiliary tag set for common core features
Member of classes:
DTD file: teicorp2
Data description:
Occurs within: settingDesc
May contain: activity locale name p time
Declaration:

```
<!ELEMENT setting       - -  (p+ | (name | time | locale |
                              activity)* )                    >

<!ATTLIST setting             %a.global
          who                 IDREFS              #IMPLIED    >
```

Discussed in: 23.2.3 ('The Setting Description') on p. 549

settingDesc

<settingDesc> (i.e. setting description) describes the setting or settings within which a language interaction takes place, either as a prose description or as a series of <setting> elements.

Attributes: [None: global and inherited attributes only.]
Example:

```
<settingDesc>
<p>Texts recorded in the Canadian Parliament building
in Ottawa, between April and November 1988
</settingDesc>
```

Part: auxiliary tag set for corpora and collections
Member of classes: declarable
DTD file: teicorp2
Data description: May contain a prose description organized as paragraphs, or a series of <setting> elements.
Occurs within: profileDesc
May contain: p setting
Declaration:

```
<!ELEMENT settingDesc   - O  (p+ | setting+)                  >

<!ATTLIST settingDesc         %a.global
                              %a.declarable                   >
```

Discussed in: 23.2 ('Contextual Information') on p. 540; 5.4 ('The Profile Description') on p. 108

settlement

<settlement> contains the name of the smallest component of a place name expressed as a hierarchy of geo-political or administrative units as in "Rochester", New York; "Glasgow", Scotland.

Attributes: [None: global and inherited attributes only.]
Example:

```
<placeName>
  <settlement type=town>Glasgow</settlement>
  <region>Scotland</region>
</placeName>
```

Part: additional tag set for names and dates
Member of classes: placePart [and indirectly also:] names
DTD file: teind2
Occurs within: placeName
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT settlement    - -  (%phrase.seq;)                   >
```

```
<!ATTLIST settlement          %a.global
                              %a.placePart                    >
```

Discussed in: 20.2 ('Place Names') on p. 493

<shift> (i.e. Shift) marks the point at which some paralinguistic feature of a series of utterances by any one speaker changes. shift

Attributes:

who supplies an identifier for the speaker or group of speakers whose shift in some feature is being noted. Its value is the identifier of a <participant> or <participant.grp> element in the TEI header.
Data type: IDREF
Value: Must identify a participant or participant group within the TEI Header
Default: #IMPLIED

feature (i.e. feature) a paralinguistic feature.
Data type: (TEMPO | LOUD | PITCH | TENSION | RHYTHM | VOICE)
Sample values include:

tempo speed of utterance.
loud loudness.
pitch pitch range.
tension tension or stress pattern.
rhythm rhythmic qualities.
voice voice quality.

Default: #REQUIRED

new (i.e. new state) specifies the new state of the paralinguistic feature specified.
Data type: CDATA
Value: An open list (for an example of possible values, see 11.3.2 ('Synchronization and Overlap') on p. 262)
Default: NORMAL
If no value is specified, it is assumed that the feature concerned ceases to be remarkable. The value "normal" has the same effect.

Example:

```
<u who=LB><shift feature=loud new=f>Elizabeth
<u who=EB>Yes
<u who=LB><shift>Come and try this <pause>
     <shift feature=loud new=ff>come on
```

The word "Elizabeth" is spoken loudly, the words "Yes" and "Come and try this" with normal volume, and the words "come on" very loudly.

Part: base tag set for spoken materials
Member of classes: comp.spoken
DTD file: teispok2
Data description: empty
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage u view
May contain: [none]
Declaration:

```
<!ELEMENT shift         - O  EMPTY                              >
<!ATTLIST shift              %a.global
          who                IDREF                #IMPLIED
          feature            (tempo | loud | pitch | tension |
                             rhythm | voice)      #REQUIRED
          new                CDATA                normal     >
```

Discussed in: 11.2.7 ('Formal Definition') on p. 259; 11.2.6 ('Shifts') on p. 258; 11.2 ('Elements Unique to Spoken Texts') on p. 253

<sic> contains text reproduced although apparently incorrect or inaccurate. sic

Attributes:

corr (i.e. correction) gives a correction for the apparent error in the copy text.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED
Example:

```
for his nose was as sharp as a pen, and
<sic corr="a' babbled">a Table</sic>
of green fields.
```

resp (i.e. responsibility) signifies the editor or transcriber responsible for suggesting the correction held as the value of the corr attribute.
Data type: IDREF
Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).
Default: %INHERITED
Example:

```
The address is Southmoor <expan abbrev='Rd' resp=LB>Road</>.
```

This attribute has no meaning if no corr attribute is supplied.

cert (i.e. certainty) signifies the degree of certainty ascribed to the correction held as the value of the corr attribute.
Data type: CDATA
Default: #IMPLIED
This attribute has no meaning if no corr attribute is supplied.

If all that is desired is to call attention to the apparent problem in the copy text, no attributes are required:

Example:

```
I don't know, Juan.  It's so far in the past now &mdash;
how <sic>we can</sic> prove or disprove anyone's theories?
```

It is also possible to provide a correct reading and to identify the individual responsible for the correction:

Example:

```
I don't know, Juan.  It's so far in the past now &mdash;
how <sic corr='can we' resp='MSM'>we can</sic> prove or
disprove anyone's theories?
```

The <sic> tag is a mirror of <corr>: the former leaves the original text untouched, giving the correction as an attribute value; the latter substitutes the correction, leaving the original reading as an attribute value. The choice between them is up to the encoder.

Part: additional tag set for common core features
Member of classes: edit [and indirectly also:] phrase
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph entry entryFree event expan eTree figure foreign formula gap gi gloss graph handShift hi kinesic l label lang lg lg1 link list listBibl m measure mentioned move name note num orig oRef oVar p pause phr ptr pRef pVar q quote ref reg rs s seg shift sic sound soCalled sp stage superentry table tag tech term termEntry text time timeRange timeStruct title tree u val view vocal w writing xptr xref TEI...end

Declaration:

```
<!ELEMENT sic           - -  (%specialPara;)                  >

<!ATTLIST sic                %a.global
          corr               CDATA               #IMPLIED
          resp               IDREF               %INHERITED

          cert               CDATA               #IMPLIED      >
```

Discussed in: 6.5.1 ('Correction of Apparent Errors') on p. 141

signed

`<signed>` (i.e. signature) contains the closing salutation, etc., appended to a foreword, dedicatory epistle, or other division of a text.

Attributes: [None: global and inherited attributes only.]

Example:

```
<signed>Thine to command <name>Humph. Moseley</name></signed>
```

Part: base tag set for common core features

Member of classes: divbot, divtop

DTD file: `teistr2`

Occurs within: back body castList closer div div0 div1 div2 div3 div4 div5 div6 div7 epilogue front group lg opener performance prologue

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT signed        - O  (%phrase.seq;)                   >

<!ATTLIST signed             %a.global                        >
```

Discussed in: 7.2.4 ('Content of Textual Divisions') on p. 194; 7.2.2 ('Openers and Closers') on p. 192

soCalled

`<soCalled>` (i.e. so called) contains a word or phrase for which the author or narrator indicates a disclaiming of responsibility, for example by the use of scare quotes or italics.

Attributes: [None: global and inherited attributes only.]

Example:

```
To edge his way along the crowded paths of life, warning
all human sympathy to keep its distance, was what the
knowing ones call <soCalled>nuts</soCalled> to Scrooge.
```

Part: base tag set for common core features

Member of classes: hqphrase [and indirectly also:] phrase

DTD file: `teicore2`

Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage

tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT soCalled      - -  (%phrase.seq;)                    >
<!ATTLIST soCalled           %a.global                         >
```

Discussed in: 6.3.3 ('Quotation') on p. 127

socecStatus

<socecStatus> (i.e. socio-economic status) contains an informal description of a person's perceived social or economic status.

Attributes:

scheme identifies the classification system or taxonomy in use.
Data type: IDREF
Value: Must identify a <taxonomy> element
Default: #IMPLIED

code identifies a status code defined within the classification system or taxonomy defined by the source attribute.
Data type: IDREF
Value: Must identify a <category> element
Default: #IMPLIED

Example:

```
<socecStatus scheme=RG code=AB1>
```

Example:

```
<socecStatus>Status AB1 in the RG Classification scheme
```

The content of this element may be used as an alternative to the more formal specification made possible by its attributes; it may also be used to supplement the formal specification with commentary or clarification.

Part: auxiliary tag set for corpora and collections
Member of classes: demographic
DTD file: teicorp2
Data description: May contain character data and phrase-level elements.
Occurs within: person personGrp

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT socecStatus   - O  (%phrase.seq)                     >
<!ATTLIST socecStatus        %a.global
          scheme             IDREF              #IMPLIED
          code               IDREF              #IMPLIED       >
```

Discussed in: 23.2.2 ('The Participants Description') on p. 545

sound

<sound> describes a sound effect or musical sequence specified within a screen play or radio script.

Attributes:

type categorizes the sound in some respect, e.g. as music, special effect, etc.
Data type: CDATA
Value: any string of characters
Default: #IMPLIED

discrete indicates whether the sound overlaps the surrounding speeches or interrupts them.
Data type: (Y | N | U)
Sample values include:

y the sound is heard between the surrounding speeches
n the sound overlaps the surrounding speeches
u unknown or inapplicable

Default: U

Example:

```
<sp><speaker>Benjy</speaker><p>Now to business.
<sp><speaker>Ford and Zaphod</speaker><p>To business.
<sound discrete=y>Glasses clink.
<sp><speaker>Benjy<speaker><p>I beg your pardon?
<sp><speaker>Ford</speaker><p>I'm sorry, I thought you were
  proposing a toast.
```

A specialized form of stage direction.

Part: base tag set for performance texts
Member of classes: stageDirection [and indirectly also:] comp.drama, inter
DTD file: teidram2
Data description: Contains character data and phrase level elements.
Occurs within: add admin argument body camera caption case castList cell colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT sound          - O  (%paraContent)                    >

<!ATTLIST sound               %a.global
          type                CDATA               #IMPLIED
          discrete            (y | n | u)         u              >
```

Discussed in: 10.3.1 ('Technical Information') on p. 248; 10.3 ('Other Types of Performance Text') on p. 246

sourceDesc

<sourceDesc> supplies a bibliographic description of the copy text(s) from which an electronic text was derived or generated.

Attributes: [None: global and inherited attributes only.]
Example:

```
<sourceDesc>
<p>No source: created in machine-readable form.</p>
</sourceDesc>
```

Part: auxiliary tag set for TEI headers
Member of classes: declarable
DTD file: teihdr2
Data description:
Occurs within: biblFull fileDesc
May contain: bibl biblFull biblStruct listBibl p recordingStmt scriptStmt
Declaration:

```
<!ELEMENT sourceDesc     - -  (p | bibl | biblFull | biblStruct
                               | listBibl | scriptStmt |
                               recordingStmt)+                >

<!ATTLIST sourceDesc          %a.global
                              %a.declarable                  >
```

Discussed in: 5.2.7 ('The Source Description') on p. 90

sp

<sp> (i.e. speech) An individual speech in a performance text, or a passage presented as such in a prose or verse text.

Attributes:

who identifies the speaker of the part by supplying an ID.
Data type: IDREFS
Value: The IDREFS are derived from the ID attribute on the role elements in the cast list or from a list of the participants.
Default: #IMPLIED

Example:

```
<sp><speaker>The reverend Doctor Opimiam</speaker>
<p>I do not think I have named a single unpresentable fish.
<sp><speaker>Mr Gryll</speaker>
<p>Bream, Doctor: there is not much to be said for bream.
<sp><speaker>The Reverend Doctor Opimiam</speaker>
<p>On the contrary, sir, I think there is much to be said for him.
In the first place....
<p>Fish, Miss Gryll -- I could discourse to you on fish by the hour:
but for the present I will forbear...
</sp>
```

The who attribute on this element may be used either in addition to the <speaker> element or as an alternative.

Part: base tag set for common core features
Member of classes: chunk [and indirectly also:] common
DTD file: teicore2
Data description: Lines or paragraphs, stage directions, and phrase-level elements.
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage view
May contain: l lg p seg speaker stage
Declaration:

```
<!ELEMENT sp            - O  (speaker?, (p | l | lg | seg |
                              stage)+)                        >

<!ATTLIST sp                 %a.global
          who                IDREFS              #IMPLIED    >
```

Discussed in: 6.11.2 ('Core Tags for Drama') on p. 179; 6.11 ('Passages of Verse or Drama') on p. 176; 10.2.2 ('Speeches and Speakers') on p. 237

space

<space> indicates the location of a significant space in the copy text.

Attributes:

dim (i.e. dimension) indicates whether the space is horizontal or vertical.
Data type: (HORIZONTAL | VERTICAL)
Sample values include:

horizontal the space is horizontal.

vertical the space is vertical.

Default: #IMPLIED
For irregular shapes in two dimensions, the value for this attribute should reflect the more important of the two dimensions. In conventional left-right scripts, a space with both vertical and horizontal components should be classed as "vertical".

extent indicates approximately how large the space is, in letters, minims, inches, or other appropriate unit.
Data type: CDATA
Value: any measured quantity, e.g. "10 letters" or "4 lines".
Default: #IMPLIED

resp incidates the individual responsible for identifying and measuring the space.
Data type: CDATA
Value: usually the initials of the responsible individual.
Default: #IMPLIED

Example:

```
<![ CDATA [
By god if wommen had writen storyes
As <space extent='7'> han within her oratoryes
```

This element should be used wherever it is desired to record an unusual space in the source text, e.g. space left for a word to be filled in later, for later rubrication, etc. It is not intended to be used to mark normal inter-word space or the like.
Part: additional tag set for transcription of primary sources
Member of classes:
DTD file: teitran2
Data description: Empty.
Occurs within: [none]
May contain: [none]
Declaration:

```
<!ELEMENT space         - O  EMPTY                              >

<!ATTLIST space              %a.global
          dim                (horizontal | vertical)
                                                  #IMPLIED
          extent             CDATA                #IMPLIED
          resp               CDATA                #IMPLIED      >
```

Discussed in: 18.2.5 ('Space') on p. 463

(i.e. span) associates an interpretative annotation directly with a span of text. span

Attributes:

value identifies the specific phenomenon being annotated.
Data type: CDATA
Value: Any string of characters.
Default: #REQUIRED

from specifies the beginning of the passage being annotated; if not accompanied by a to attribute, then specifies the entire passage.
Data type: IDREF
Value: The ID of the element which occurs at the beginning of the passage.
Default: #REQUIRED

to specifies the end of the passage being annotated.
Data type: IDREF
Value: The ID of the element which occurs at the end of the passage.
Default: #IMPLIED

Example:

```
<span resp=TMA type='structural unit' value='aftermath' from=P2 to=P4>
```

Part: additional tag set for
Member of classes: interpret, metadata [and indirectly also:] globincl
DTD file: teiana2
Data description: Empty element.
Occurs within: spanGrp
May contain: [none]
Declaration:

```
<!ELEMENT span          - O  EMPTY                              >

<!ATTLIST span               %a.global
                             %a.interpret
```

```
            value           CDATA           #REQUIRED
            from            IDREF           #REQUIRED
            to              IDREF           #IMPLIED      >
```

Discussed in: 15.3 ('Spans and Interpretations') on p. 387

spanGrp

<spanGrp> (i.e. span group) collects together tags.

Attributes: [None: global and inherited attributes only.]
Example:

```
<spanGrp resp=TMA type='collection of structural units'>
 <span value='introduction' from=S1 to=S3>
 <span value='conflict' from=S4a>
 <span value='climax' from=S4b>
 <span value='revenge' from=S5 to=Sx>
 <span value='reconciliation' from=nil1>
 <span value='aftermath' from=P2 to=P4>
 </spanGrp>
```

Part: additional tag set for
Member of classes: interpret, metadata [and indirectly also:] globincl
DTD file: teiana2
Data description: Any number of elements.
Occurs within: [none]
May contain: span
Declaration:

```
<!ELEMENT spanGrp       - -  ((span)*)                    >

<!ATTLIST spanGrp            %a.global
                             %a.interpret                 >
```

Discussed in: 15.3 ('Spans and Interpretations') on p. 387

speaker

<speaker> A specialized form of heading or label, giving the name of one or more speakers in a dramatic text or fragment.

Attributes: [None: global and inherited attributes only.]
Example:

```
<sp who="NI RSA">
<speaker>Nancy and Robert</speaker>
<stage type=delivery>(speaking simultaneously)</stage>
<p>The future? ...
</sp>
```

The who attribute on the <sp> element may be used either in addition to this element or as an alternative.
Part: base tag set for common core features
Member of classes:
DTD file: teicore2
Data description: Any sequence of phrase level data
Occurs within: sp [May not appear at any level within: speaker]
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref [May not include at any level: speaker]
Declaration:

```
<!ELEMENT speaker       - O  (%phrase.seq)        -(speaker)    >

<!ATTLIST speaker            %a.global                        >
```

Discussed in: 6.11.2 ('Core Tags for Drama') on p. 179

sponsor

<sponsor> specifies the name of a sponsoring organization or institution.

Attributes: [None: global and inherited attributes only.]
Example:

```
<sponsor>Association for Computers and the Humanities
</sponsor>
<sponsor>Association for Computational Linguistics
</sponsor>
<sponsor>Association for Literary and Linguistic Computing
</sponsor>
```

Sponsors give their intellectual authority to a project; they are to be distinguished from *funders*, who provide the funding but do not necessarily take intellectual responsibility.
Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:
Occurs within: titleStmt
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT sponsor       - O  ( %phrase.seq; )                >
<!ATTLIST sponsor            %a.global                       >
```

Discussed in: 5.2.1 ('The Title Statement') on p. 82

<stage> (i.e. stage direction) contains any kind of stage direction within a dramatic text or fragment. stage

Attributes:

type indicates the kind of stage direction.
Data type: CDATA
Sample values include:

setting describes a setting.
entrance describes an entrance.
exit describes an exit.
business describes stage business.
novelistic is a narrative, motivating stage direction.
delivery describes how a character speaks.
modifier gives some detail about a character.
location describes a location.
mixed more than one of the above

Default: MIX

Example:

```
<stage type="setting">A curtain being drawn.</stage>
<stage type="setting">Music</stage>
<stage type="entrance">Enter Husband as being thrown off his
horse.</stage>
<stage type="exit">Exit pursued by a bear.</stage>
<stage type="business">He quickly takes the stone out.</stage>
<stage type="delivery">To Lussurioso.</stage>
<stage type="novelistic">Having had enough, and embarrassed for
the family.</stage>
<stage type="modifier">Disguised as Ansaldo.</stage>
<stage type="location">At a window.</stage>
<stage type="delivery rend="inline">Aside.</stage>
```

Part: base tag set for common core features
Member of classes: common, inter

DTD file: teicore2

Occurs within: add admin argument body camera caption case castList cell colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound sp stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref [May not appear at any level within: stage state]

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph entry entryFree event expan eTree figure foreign formula gap gi gloss graph handShift hi kinesic l label lang lg lg1 link list listBibl m measure mentioned move name note num orig oRef oVar p pause phr ptr pRef pVar q quote ref reg rs s seg shift sic sound soCalled sp stage superentry table tag tech term termEntry text time timeRange timeStruct title tree u val view vocal w writing xptr xref TEI...end [May not include at any level: stage]

Declaration:

```
<!ELEMENT stage          - -  (%specialPara)       -(stage)      >

<!ATTLIST stage                %a.global
          type                 CDATA                mix           >
```

Discussed in: 6.11.2 ('Core Tags for Drama') on p. 179; 6.11 ('Passages of Verse or Drama') on p. 176; 10.2.3 ('Stage Directions') on p. 238

state

<state> specifies one component of a canonical reference defined by the "milestone" method.

Attributes:

ed (i.e. edition) indicates which edition or version the milestone applies to.
Data type: CDATA
Value: Any string of characters; usually a siglum conventionally used for the edition.
Default: #IMPLIED
If ed is not specified, then any milestone tag with an appropriate unit attribute will be selected.

unit indicates what kind of section is changing at this milestone.
Data type: CDATA
Sample values include:

page page breaks in the reference edition.
column column breaks.
line line breaks.
book any units termed "book","liber", etc.
poem individual poems in a collection.
canto cantos or other major sections of a poem.
stanza stanzas within a poem, book, or canto.
act acts within a play.
scene scenes within a play or act.
section sections of any kind.
absent passages not present in the reference edition.

Default: #REQUIRED

length specifies the fixed length of the reference component.
Data type: NUMBER
Value: Should be a positive integer; if no value is provided, the length is unlimited and goes to the next delimiter or to the end of the value.
Default: #IMPLIED
When constructing a reference, if the reference component found is of numeric type, the length is made up by inserting leading zeros; if it is not, by inserting trailing blanks. In either case, reference components are truncated if necessary at the right hand side.
When seeking a reference, the length indicates the number of characters which should be compared. Values longer than this will be regarded as matching, if they start correctly.

delim supplies a delimiting string following the reference component.
Data type: CDATA
Value: If a single space is used it is interpreted as whitespace.
Default: #IMPLIED

Example:

```
<state unit=book delim=":">
<state unit=line length=4>
```

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:
Occurs within: refsDecl
May contain: [none] [May not include at any level: stage]
Declaration:

```
<!ELEMENT state          - O  EMPTY                              >

<!ATTLIST state               %a.global
          ed                  CDATA                #IMPLIED
          unit                CDATA                #REQUIRED
          length              NUMBER               #IMPLIED
          delim               CDATA                #IMPLIED      >
```

Discussed in: 5.3.5.3 ('Milestone Method') on p. 102; 5.3.5 ('The Reference System Declaration ') on p. 100

<stdVals> (i.e. Standard values) specifies the format used when standardized date or number values are supplied. stdVals

Attributes: [None: global and inherited attributes only.]
Example:

```
<stdVals>
  <p>Dates are represented in ISO order: YYYYMMDD.
  <p>All integer numbers are left-filled to 8 digits.
</stdVals>
```

Part: auxiliary tag set for TEI headers
Member of classes: declarable
DTD file: teihdr2
Data description:
Occurs within: editorialDecl
May contain: p
Declaration:

```
<!ELEMENT stdVals        - O  (p+)                               >

<!ATTLIST stdVals             %a.global
                              %a.declarable                     >
```

Discussed in: 5.3.3 ('The Editorial Practices Declaration') on p. 96; 23.3.2 ('Declarable Elements') on p. 552

<step> (i.e. step) specifies one component of a canonical reference defined by the "stepwise" method. step

Attributes:

refunit (i.e. reference unit) names the unit (book, chapter, canto, verse, ...) identified by this step in a canonical reference.
Data type: CDATA
Value: any string of characters; typically a word or phrase in some natural language.
Default: #IMPLIED
The provision of this attribute helps make the structure of the canonical reference much clearer. Its use is strongly recommended.

length specifies the fixed length of the reference component.
Data type: NUMBER
Value: Should be a positive integer; if no value is provided, the length is unlimited and goes to the next delimiter or to the end of the value.
Default: #IMPLIED
When constructing a reference, if the reference component found is of numeric type, the length is made up by inserting leading zeros; if it is not, by inserting trailing blanks. In either case, reference components are truncated if necessary at the right hand side.
When seeking a reference, the length indicates the number of characters which should be compared. Values longer than this will be regarded as matching, if they start correctly.

delim supplies a delimiting string following the reference component.
Data type: CDATA
Value: If a single space is used it is interpreted as whitespace
Default: #IMPLIED

from specifies the starting point of the area referred to by this step in the canonical reference.
Data type: %EXTPTR
Value: a valid expression in the TEI extended pointer notation documented in section 14.2 ('Extended Pointers') on p. 340.
Default: #REQUIRED

to specifies the ending point of the area referred to by this step in the canonical reference.
Data type: %EXTPTR
Value: a valid expression in the TEI extended pointer notation documented in section 14.2 ('Extended Pointers') on p. 340.
Default: 'DITTO'
(optional)

Example:

```
<step delim=":" from='CHILD (1 DIV1 N %1)' refunit='book'>
```

With two differences, the extended pointer expressions given as values for the from and to attributes behave like those given for the corresponding attributes of an extended pointer element like <xptr>. First: unlike normal extended pointer expressions, those on a <step> element can contain references to tokens in the canonical reference string itself, expressed as %1, %2, etc. Second: the initial location source for the first step is the root of the document, as for normal extended pointers. The initial location source for subsequent steps, however, is the result of the previous step, not the root.

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:
Occurs within: refsDecl
May contain: [none]
Declaration:

```
<!ELEMENT step          - O  EMPTY                          >
<!ATTLIST step               %a.global
          refunit            CDATA              #IMPLIED
          length             NUMBER             #IMPLIED
          delim              CDATA              #IMPLIED
          from               %extPtr            #REQUIRED
          to                 %extPtr            'DITTO'     >
```

Discussed in: 5.3.5.3 ('Milestone Method') on p. 102; 5.3.5 ('The Reference System Declaration ') on p. 100; 5.3.5.2 ('Stepwise Method') on p. 101

str

<str> (i.e. String value) provides a string value for a feature.

Attributes:

rel indicates the relation of the given value to the actual value.
Data type: (EQ | NE | SB | NS | LT | LE | GT | GE)

Sample values include:

eq indicates that the actual value is that given.
ne indicates that the actual value is not that given.
sb indicates that the value given is a substring of the actual value.
ns indicates that the value given is not a substring of the actual value.
lt indicates that the actual value is less than the given value.
le indicates that the actual value is less than or equal to the given value.
gt indicates that the actual value is greater than the given value.
ge indicates that the actual value is greater than or equal to the given value.

Default: EQ

The use of rel=lt, etc. assumes that an ordering of string values has been defined.

Example:

```
<str>Hello, world!</str>
```

Part: additional tag set for feature structures
Member of classes: singleVal [and indirectly also:] featureVal
DTD file: teifs2
Data description: Parsed character data.
Occurs within: f fvLib if vAlt vDefault vRange
May contain: #PCDATA
Declaration:

```
<!ELEMENT str           - -  (#PCDATA)                          >

<!ATTLIST str                %a.global
          rel                (eq | ne | sb | ns | lt | le | gt
                             | ge)               eq           >
```

Discussed in: 16.4 ('Symbolic, Numeric, Measurement, Rate and String Values') on p. 402

<street> (i.e. street) a full street address including any name or number identifying a building as well as the name of the street or route on which it is located. street

Attributes: [None: global and inherited attributes only.]
Example:

```
<street>via della Faggiola, 36</street>
```

Example:

```
<street>Duntaggin,
        110 Southmoor Road
</street>
```

The order and presentation of house names and numbers and street names, etc., may vary considerably in different countries. The encoding should reflect the order which is appropriate in the country concerned.

Part: base tag set for common core features
Member of classes: addrPart
DTD file: teicore2
Data description: Running prose only
Occurs within: address
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT street        - O  (%phrase.seq)                      >

<!ATTLIST street             %a.global                          >
```

Discussed in: 6.4.2 ('Addresses') on p. 134

<stress> contains the stress pattern for a dictionary headword, if given separately. stress

Attributes: [None: global and inherited attributes only.]
Part: base tag set for printed dictionaries
Member of classes: dictionaryParts
DTD file: teidict2
Data description: May contain character data and phrase-level elements.
Occurs within: trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT stress        - O  (%paraContent;)                >

<!ATTLIST stress             %a.global                      >
```

Discussed in: 12.3.1 ('Information on Written and Spoken Forms') on p. 279

string

<string> contains the intended expansion for the entity documented by an <entdoc> element, enclosed by quotation marks.

Attributes: [None: global and inherited attributes only.]
Example:

```
<string>"the choice of quotes may'nt be insignificant"</string>
```

System entities should include the *system* keyword within the content of this element, as shown:
Example:

```
<string>SYSTEM 'teiclas2.ent'</>
```

The content of this element is the SGML *entity text* for the entity, including any keywords.
Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: any sequence of character data
Occurs within: entDoc
May contain: #PCDATA
Declaration:

```
<!ELEMENT string        - -  (#PCDATA)                      >

<!ATTLIST string             %a.global                      >
```

Discussed in: 27.3 ('Entity Documentation ') on p. 607

subc

<subc> (i.e. subcategorization) contains subcategorization information (transitive/intransitive, countable/non-count, etc.)

Attributes: [None: global and inherited attributes only.]
Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, gramInfo
DTD file: teidict2
Data description: May contain character data and phrase-level elements.
Occurs within: gramGrp trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT subc          - O  (%paraContent;)                >

<!ATTLIST subc               %a.global
                             %a.dictionaries                >
```

Discussed in: 12.3.2 ('Grammatical Information') on p. 284

<superentry> groups successive entries for a set of homographs.

Attributes: [None: global and inherited attributes only.]
Part: base tag set for printed dictionaries
Member of classes: comp.dictionaries, dictionaryParts, entries
DTD file: teidict2
Data description: May contain character data and phrase-level elements.
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage trans view
May contain: entry form
Declaration:

```
<!ELEMENT superentry    - O  (form?, entry+)                >
<!ATTLIST superentry         %a.global
                             %a.entries                     >
```

Discussed in: 12.1 ('Dictionary Body and Overall Structure') on p. 270

<supplied> signifies text supplied by the transcriber or editor in place of text which cannot be read, either because of physical damage or loss in the original or because it is illegible for any reason. supplied

Attributes:

reason indicates why the text has had to be supplied.
Data type: CDATA
Value: any phrase describing the difficulty, e.g. "overbinding", "faded ink", "lost folio", "omitted in original".
Default: #IMPLIED

resp indicates the individual responsible for supplying the letter, word or passage contained within the <supplied> element.
Data type: CDATA
Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).
Default: %INHERITED

hand Where the presumed loss of text leading to the supplying of text arises from action (partial deletion, etc.) assignable to an identifiable hand, signifies the hand responsible for the action.
Data type: IDREF
Value: must be one of the hand identifiers declared in the document header (see section 18.2.1 ('Document Hands') on p. 456).
Default: %INHERITED

agent where the presumed loss of text leading to the supplying of text arises from an identifiable cause, signifies the causative agent.
Data type: CDATA
Value: any prose description of the agent.
Default: #IMPLIED

source states the source of the supplied text.
Data type: CDATA
Value: any string of characters identifying the source of the supplied text. This might be the sigil for a manuscript, or a particular edition, or the transcriber or editor's own initials, indicating it as their conjecture.
Default: #IMPLIED

Example:

```
I am dr Sr yr
<supplied reason='illegible'
          resp='RW'
```

```
            source='amanuensis copy'>very humble Servt</>
    Sydney Smith
```

The <damage>, <omit>, <del>, <unclear> and <supplied> elements may be closely allied in use. See section 18.2.4 ('The Use of the Gap, Del, Damage, Unclear and Supplied Tags in Combination') on p. 462 for discussion of which element is appropriate for which circumstance.
Part: additional tag set for text criticism
Member of classes:
DTD file: teitran2
Data description: May contain character data and phrase-level elements.
Occurs within: [none]
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT supplied      - O  (%paraContent;)                    >

<!ATTLIST supplied           %a.global
          reason             CDATA               #IMPLIED
          resp               CDATA               %INHERITED

          hand               IDREF               %INHERITED

          agent              CDATA               #IMPLIED
          source             CDATA               #IMPLIED      >
```

Discussed in: 18.2.3 ('Damage, Illegibility, and Supplied Text') on p. 460

surname

<surname> (i.e. surname) contains a family (inherited) name, as opposed to a given, baptismal, or nick name.
Attributes: [None: global and inherited attributes only.]
Example:

```
<surname type=combine>St John Stevas</surname>
```

Part: additional tag set for names and dates
Member of classes: personPart [and indirectly also:] names
DTD file: teind2
Occurs within: persName
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT surname       - -  (%phrase.seq;)                     >

<!ATTLIST surname            %a.global
                             %a.personPart                     >
```

Discussed in: 20.1 ('Personal Names') on p. 488

syll

<syll> (i.e. syllabification) contains the syllabification of the headword.
Attributes: [None: global and inherited attributes only.]
Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, formInfo
DTD file: teidict2
Data description: May contain character data and phrase-level elements.
Occurs within: form trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref

Declaration:

```
<!ELEMENT syll          - O  (%paraContent;)                  >
<!ATTLIST syll               %a.global
                             %a.dictionaries                  >
```

Discussed in: 12.3.1 ('Information on Written and Spoken Forms') on p. 279

<sym> (i.e. Symbolic value) provides symbolic values for features. sym

Attributes:

value provides a symbolic value for a feature, one of a finite list that may be specified in a feature declaration.
Data type: CDATA
Value: A string, e.g. *feminine*.
Default: #REQUIRED

rel indicates the relation of the given value to the actual value.
Data type: (EQ | NE)
Sample values include:
eq indicates that the actual value is that given.
ne indicates that the actual value is not that given.
Default: EQ

Example:

```
<sym rel=ne value=feminine>
```

Part: additional tag set for feature structures
Member of classes: singleVal [and indirectly also:] featureVal
DTD file: teifs2
Data description: Empty tag.
Occurs within: f fvLib if vAlt vDefault vRange
May contain: [none]
Declaration:

```
<!ELEMENT sym           - O  EMPTY                            >
<!ATTLIST sym                %a.global
          value              CDATA              #REQUIRED
          rel                (eq | ne)          eq            >
```

Discussed in: 16.4 ('Symbolic, Numeric, Measurement, Rate and String Values') on p. 402

<symbol> documents the intended significance of a particular character or character sequence within a metrical notation, either explicitly or in terms of other <symbol> elements in the same <metNotation>. symbol

Attributes:

value specifies the character or character sequence being documented.
Data type: CDATA
Value: any available character or character sequence.
Default: #REQUIRED

terminal specifies whether the symbol is defined in terms of other symbols (terminal=N) or in prose (terminal=Y).
Data type: (Y | N)
Sample values include:
Y the element contains a prose definition of its meaning.
N the element contains a definition of its meaning given using symbols defined elsewhere in the same <metNotation> element.
Default: Y

Example:

```
<symbol value='x'>a stressed syllable
<symbol value='o'>an unstressed syllable
<symbol value='A' terminal=N>xoo
```

Part: base tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description: May contain character data and phrase-level elements.
Occurs within: metDecl
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT symbol        - O  (%phrase.seq;)                    >

<!ATTLIST symbol             %a.global
          value              CDATA              #REQUIRED
          terminal           (Y | N)            Y              >
```

Discussed in: 5.3.8 ('The Metrical Declaration Element') on p. 106; 9.4 ('Rhyme and Metrical Analysis') on p. 221

table

<table> contains text displayed in tabular form, in rows and columns.

Attributes:

rows indicates the number of rows in the table.
Data type: NUMBER
Value: If no number is supplied, an application must calculate the number of rows.
Default: #IMPLIED
Rows should be presented from top to bottom.

cols indicates the number of columns in each row of the table.
Data type: NUMBER
Value: If no number is supplied, an application must calculate the number of columns.
Default: #IMPLIED
Within each row, columns should be presented left to right.

Example:

```
<table rows=4 cols=4>
<head>Poor Men's Lodgings in Norfolk (Mayhew, 1843)
<row role=label>
  <cell></>
  <cell>Dossing Cribs or Lodging Houses</>
  <cell>Beds</>
  <cell>Needys or Nightly Lodgers</></row>
<row role=data>
  <cell role=label>Bury St Edmund's</>
  <cell>5</><cell>8</><cell>128</></row>
<row role=data>
  <cell role=label>Thetford</>
  <cell>3</><cell>6</><cell>36</></row>
<row role=data>
  <cell role=label>Attleboro'</>
  <cell>3</><cell>5</><cell>20</></row>
<row role=data>
  <cell role=label>Wymondham</>
  <cell>1</><cell>11</><cell>22</></row>
<!-- ... -->
</table>
```

Any rendition information should be supplied using the global rend attribute, at the table, row, or cell level as appropriate.
Part: base tag set for formulae
Member of classes: inter
DTD file: teifig2
Data description: Contains an optional heading and a series of rows.

Occurs within: add admin camera caption case cell colloc corr country damage def desc descrip docEdition emph equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting mood note number orth otherForm p per pos pron q quote rdg ref region rendition row seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref

May contain: head row

Declaration:

```
<!ELEMENT table          - -  (head*, row+)                    >

<!ATTLIST table               %a.global
          rows                NUMBER              #IMPLIED
          cols                NUMBER              #IMPLIED      >
```

Discussed in: 22.1.1 ('The TEI Table DTD') on p. 524

<tag> contains text of a complete SGML start- or end-tag, possibly including attribute specifications, but excluding the opening and closing markup delimiter characters. tag

Attributes:

TEI indicates whether this tag is valid within the TEI scheme or not.
Data type: (YES | NO)
Sample values include:
yes this is a valid TEI tag.
no this is not a valid TEI tag.
Default: YES

Example:

```
Mark the start of each italicised phrase with a <tag>hi rend=it</tag>
tag, and its end with a <tag>/hi</tag> tag.
```

Part: auxiliary tag set for tag set declarations

Member of classes: sgmlKeywords [and indirectly also:] phrase

DTD file: teitsd2

Data description: contains only characters legal between markup delimiter characters.

Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: #PCDATA

Declaration:

```
<!ELEMENT tag            - -  (#PCDATA)                        >

<!ATTLIST tag                 %a.global
          TEI                 (yes | no)          yes           >
```

Discussed in: 27 ('Tag Set Documentation') on p. 601

<tagDoc> documents the structure, content, and purpose of a single SGML element type. tagDoc

Attributes:

usage specifies the optionality of an attribute or element.
Data type: (REQ | MWA | REC | RWA | OPT)
Sample values include:
req required

mwa mandatory when applicable
rec recommended
rwa recommended when applicable
opt optional
Default: OPT

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description:
Occurs within: tsd
May contain: attlDecl attList children classes dataDesc desc elemDecl equiv exemplum files gi parents part ptr remarks rs
Declaration:

```
<!ELEMENT tagDoc        - -  (gi, rs?, desc, attList?,
                              exemplum*, remarks?, part?,
                              classes?, files?, dataDesc?,
                              parents?, children?, elemDecl,
                              attlDecl?, ptr*, equiv*)          >

<!ATTLIST tagDoc              %a.global
          usage               (req | mwa | rec | rwa | opt)
                                                    opt         >
```

Discussed in: 27.1 ('The TagDoc Documentation Element') on p. 603; 27 ('Tag Set Documentation') on p. 601

tagsDecl

<tagsDecl> (i.e. tagging declaration) provides detailed information about the tagging applied to an SGML document.

Attributes: [None: global and inherited attributes only.]
Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Occurs within: encodingDesc
May contain: rendition tagUsage
Declaration:

```
<!ELEMENT tagsDecl      - O  (rendition*, tagUsage*)          >

<!ATTLIST tagsDecl           %a.global                        >
```

Discussed in: 5.3.4 ('The Tagging Declaration') on p. 98; 5.3 ('The Encoding Description') on p. 93

tagUsage

<tagUsage> (i.e. tagUsage) supplies information about the usage of a specific element within a <text>.

Attributes:

gi the name (generic identifier) of the element indicated by the tag.
Data type: NAME
Value: the name of an element from the current dtd
Default: #REQUIRED

occurs specifies the number of occurrences of this element within the text.
Data type: NUMBER
Value: an integer number greater than zero
Default: #IMPLIED

ident specifies the number of occurrences of this element within the text which bear a distinct value for the global id attribute.
Data type: NUMBER
Value: an integer number greater than zero
Default: #IMPLIED

render specifies the identifier of a <rendition> element which defines how this element is to be rendered.
Data type: IDREF
Value: an SGML identifier specified as the value of the id attribute on some <rendition> element in the current document.
Default: #IMPLIED

Example:

```
<tagUsage gi=hi occurs=28 ident=2 render=IT>
    Used only to mark English words italicised in the copy text
</tagUsage>
```

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Occurs within: tagsDecl
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT tagUsage      - O  (%paraContent)                    >
<!ATTLIST tagUsage           %a.global
          gi                 NAME                #REQUIRED
          occurs             NUMBER              #IMPLIED
          ident              NUMBER              #IMPLIED
          render             IDREF               #IMPLIED       >
```

Discussed in: 5.3.4 ('The Tagging Declaration') on p. 98

taxonomy

<taxonomy> (i.e. taxonomy) defines a typology used to classify texts either implicitly, by means of a bibliographic citation, or explicitly by a structured taxonomy.

Attributes: [None: global and inherited attributes only.]
Example:

```
<taxonomy id=B>
  <bibl>Brown Corpus</bibl>
  <category id=B.A><catdesc>Press Reportage
    <category id=B.A1><catdesc>Daily</category>
    <category id=B.A2><catdesc>Sunday</category>
    <category id=B.A3><catdesc>National</category>
    <category id=B.A4><catdesc>Provincial</category>
    <category id=B.A5><catdesc>Political</category>
    <category id=B.A6><catdesc>Sports</category>
  </category>
  <category id=B.D><catDesc>Religion
    <category id=B.D1><catdesc>Books</category>
    <category id=B.D2><catdesc>Periodicals and tracts</category>
  </category>
</taxonomy>
```

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Occurs within: classDecl
May contain: bibl biblFull biblStruct category
Declaration:

```
<!ELEMENT taxonomy      - -  (category+ | ((bibl | biblStruct |
                              biblFull), category*))          >
```

```
<!ATTLIST taxonomy          %a.global                       >
```

Discussed in: 5.3.6 ('The Classification Declaration') on p. 104

tech

<tech> (i.e. Technical stage direction) describes a special purpose stage direction that is not meant for the actors.

Attributes:

type categorizes the technical stage direction.
Data type: (LIGHT | SOUND | PROP | BLOCK)
Sample values include:
light a lighting cue.
sound a sound cue.
prop a prop cue.
block a blocking instruction
Default: #IMPLIED

perf identifies the performance or performances to which this technical direction applies.
Data type: IDREFS
Value: The ID of a <performance> element.
Default: #IMPLIED

Example:

```
<tech type=light>Red spot on his face</tech>
```

Part: base tag set for performance texts
Member of classes: stageDirection [and indirectly also:] comp.drama, inter
DTD file: teidram2
Data description: Contains character level information.
Occurs within: add admin argument body camera caption case castList cell colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT tech          - O  (%paraContent)                  >

<!ATTLIST tech              %a.global
          type              (light | sound | prop | block)
                                                  #IMPLIED
          perf              IDREFS                #IMPLIED   >
```

Discussed in: 10.3.1 ('Technical Information') on p. 248

TEI.2

<TEI.2> (i.e. TEI 2 document) contains a single TEI-conformant document, comprising a TEI header and a text, either in isolation or as part of a <teiCorpus> element.

Attributes: [None: global and inherited attributes only.]
Example:

```
<TEI.2>
  <teiHeader>
    <fileDesc>
       <titleStmt>
          <title>The shortest TEI Document Imaginable</title>
       <publicationStmt>
          <p>Published as part of TEI P2.
       </publicationStmt>
```

```
      <sourceDesc>
            <p>No source: this is an original work.
      </sourceDesc>
    </teiHeader>
    <text>
      <body>
       <p>This is about the shortest TEI document imaginable.
      </body>
    </text>
  </TEI.2>
```

This element is required.
Part: auxiliary tag set for common core features
Member of classes:
DTD file: tei2
Data description: Must contain one TEI header and one text.
Occurs within: teiCorpus.2
May contain: teiHeader text
Declaration:

```
<!ELEMENT TEI.2         - O  (teiHeader, text)                  >
<!ATTLIST TEI.2              %a.global                          >
```

Discussed in: 3.6.1 ('Structure of the TEI2.DTD File') on p. 46; 23.1 ('Varieties of Composite Text') on p. 538

<teiCorpus.2> (i.e. TEI corpus) contains the whole of a TEI encoded corpus, comprising a single corpus header and one or more TEI.2 elements, each containing a single text header and a text. teiCorpus.2

Attributes: [None: global and inherited attributes only.]
Example:

```
<teiCorpus.2>
    <teiHeader> ...
    <!-- header for corpus-level information -->
    </teiHeader>
    <TEI.2 id=T1>
         <teiHeader> ...
         <!-- header for text-level information -->
         </teiHeader>
         <text> ... </text>
    </TEI.2>
    <TEI.2 id=T2>
         <teiHeader> ... </teiHeader>
         <text> ... </text>
    </TEI.2>
    <!-- etc. -->
</teiCorpus.2>
```

This element is mandatory when applicable.
Part: auxiliary tag set for common core features
Member of classes:
DTD file: tei2
Data description: Must contain one TEI header for the corpus, and a series of <TEI.2> elements, one for each text.
Occurs within: [none]
May contain: teiHeader TEI.2
Declaration:

```
<!ELEMENT teiCorpus.2   - O  (teiHeader, TEI.2+)                >
```

```
<!ATTLIST teiCorpus.2          %a.global                              >
```

Discussed in: 3.6.1 ('Structure of the TEI2.DTD File') on p. 46; 23.1 ('Varieties of Composite Text') on p. 538

teiFsd2

<teiFsd2> (i.e. feature system declaration) contains a feature system declaration.

Attributes: [None: global and inherited attributes only.]
Part: auxiliary tag set for feature system declarations
Member of classes:
DTD file: teifsd2
Data description: Must contain a standard TEI header, a set of feature declarations, and optionally a set of feature co-occurrence constraints.
Occurs within: [none]
May contain: fsDecl teiHeader
Declaration:

```
<!ELEMENT teiFsd2        - -  (teiHeader, fsDecl+)                    >
<!ATTLIST teiFsd2              %a.global                              >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

teiHeader

<teiHeader> (i.e. TEI Header) supplies the descriptive and declarative information making up an "electronic title page" prefixed to every TEI-conformant text.

Attributes:

type specifies the kind of document to which the header is attached.
Data type: CDATA
Sample values include:
text the header is attached to a single text.
corpus the header is attached to a corpus.
Default: TEXT

creator identifies the creator of the TEI Header.
Data type: CDATA
Value: The name or initials of the person or institution responsible for creating this TEI header.
Default: #IMPLIED

status indicates whether the header is new or has been substantially revised.
Data type: (NEW | UPDATE)
Sample values include:
new the header is a new header.
update the header is an update (has been revised).
Default: NEW

date.created indicates when the first version of the header was created.
Data type: CDATA
Value: A date in any recognizable form.
Default: #IMPLIED

date.updated indicates when the current version of the header was created.
Data type: CDATA
Value: A date in any recognizable form.
Default: #IMPLIED

Example:

```
<teiHeader>
<fileDesc>
   <titleStmt>
      <title>Shakespeare: the first folio (1623) in electronic form
      <author>Shakespeare, William (1564-1616)
      <resp><role>Originally prepared by</>
      <date>1968
```

```
            <name>Trevor Howard-Hill</>
         <resp><role>Revised and edited by</>
            <name>Christine Avern-Carr</>
      <publicationStmt>
         <distributor>Oxford Text Archive
         <address>13 Banbury Road, Oxford OX2 6NN, UK
         <idno>OTA number 119
         <availability>Freely available on a non-commercial basis.
         <date>1968
      <sourceDesc>
         <bibl>The first folio of Shakespeare, prepared by Charlton Hinman
         (The Norton Facsimile, 1968)</bibl>
   </fileDesc>
   <encodingDesc>
      <projectDesc>
      <p>Originally prepared for use in the production of a series of
         old-spelling concordances in 1968, this text was extensively
         checked and revised for use during the editing of the new Oxford
         Shakespeare (Wells and Taylor, 1989).
      <correction>
      <p>Turned letters are silently corrected.
      <normalization>
      <p>Original spelling and typography is retained, except that long s
         and ligatured forms are not encoded.
      </editorialDecl>
      <refsDecl id=ASLref>
         <step gi=div1 attr=n><delim string=" ">
         <step gi=div2 attr=n><delim string=".">
         <step gi=l attr=n>
      </refsDecl>
   </encodingDesc>
   <revisionDesc>
       <change><by>CAC<date>12 Apr 89<what>Last checked</change>
       <change><by>LB<date>1 Mar 89<what>Made new file</change>
   </revisionDesc>
   </teiHeader>
```

One of the few elements unconditionally required in any TEI document; the header may also be exchanged as an independent document.

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:
Occurs within: ihs teiCorpus.2 teiFsd2 TEI.2
May contain: encodingDesc fileDesc profileDesc revisionDesc
Declaration:

```
<!ELEMENT teiHeader     - -  (fileDesc, encodingDesc*,
                              profileDesc*, revisionDesc?)       >
<!ATTLIST teiHeader           %a.global
          type                CDATA                text
          creator             CDATA                #IMPLIED
          status              (new | update)       new
          date.created        CDATA                #IMPLIED
          date.updated        CDATA                #IMPLIED       >
```

Discussed in: 5.1.1 ('The TEI Header and Its Components') on p. 78; 23.1 ('Varieties of Composite Text') on p. 538

term

<term> contains a single-word, multi-word or symbolic designation which is regarded as a technical term.

Attributes:

type classifies the term using some typology.
Data type: CDATA
Value: any string of characters; for serious terminological work, values should be taken from the dictionary of data element types specified in ISO WD 12 620.
Default: #IMPLIED

Example:

```
A computational device that infers structure from grammatical
strings of words is known as a <term>parser</term>, and much
of the history of NLP over the last 20 years has been occupied
with the design of parsers.
```

Example:

```
We may define <term rend=sc id=tdpv>discoursal point of view</term>
as <gloss target=tdpv>the relationship, expressed through discourse
structure, between the implied author or some other addresser,
and the fiction.</gloss>
```

In formal terminological work, there is frequently discussion over whether terms must be atomic or may include multi-word lexical items, symbolic designations, or phraseological units. The <term> element may be used to mark any of these. No position is taken on the philosophical issue of what a term can be; the looser definition simply allows the <term> element to be used by practitioners of any persuasion.

Part: base tag set for common core features
Member of classes: hqphrase [and indirectly also:] phrase
DTD file: teicore2
Data description: free prose
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype keywords l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term termEntry tig time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT term          - -  (%phrase.seq;)                    >
<!ATTLIST term                %a.global
          type                CDATA                 #IMPLIED   >
```

Discussed in: 6.3.4 ('Terms, Glosses, and Cited Words') on p. 130; 13.2 ('Tags for Terminological Data') on p. 312

termEntry

<termEntry> (i.e. terminological entry (flat structure)) contains a single complete entry for one or more terms and their associated descriptive and administrative data.

Attributes:

type classifies the term entry using some typology, preferably the dictionary of data element types specified in ISO WD 12 620.
Data type: CDATA
Value: any string of characters; for serious terminological work, values should be taken from the dictionary of data element types specified in ISO WD 12 620.
Default: #IMPLIED

Often but not always a term entry describes all terms in a term bank denoting a given concept. In the case of a term entry documenting a concept for which no standard term exists, the term entry will have to contain an empty "dummy" <term>, of the form <term></term>.

Part: base tag set for terminological data
Member of classes: comp.terminology
DTD file: teite2f
Data description: May contain character data and phrase-level elements.
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage view
May contain: admin date dateStruct descrip gram note otherform ptr ref term xptr xref
Declaration:

```
<!ELEMENT termEntry     - O  ( (%m.terminologyMisc | otherform
                             | gram |
                             %m.terminologyInclusions)*, (term,
                             (%m.terminologyMisc | otherform |
                             gram | %m.terminologyInclusions)*
                             )+ )                              >

<!ATTLIST termEntry          %a.global
          type               CDATA               #IMPLIED       >
```

Discussed in: 13.3 ('Basic Structure of the Terminological Entry') on p. 316

termEntry

<termEntry> (i.e. terminological entry (nested structure)) contains a single complete entry for one concept expressed in one language and comprising one or more terms and their associated descriptive and administrative data, or, in bilingual and multilingual terminology work, two or more very closely related concepts comprising one or more terms in each language and their associated descriptive and administrative data.

Attributes:

type classifies the term entry using some typology, preferably the dictionary of data element types specified in ISO WD 12 620.
Data type: CDATA
Value: any string of characters; for serious terminological work, values should be taken from the dictionary of data element types specified in ISO WD 12 620.
Default: #IMPLIED

Often but not always a term entry describes all terms in a term bank denoting a given concept. In the case of a term entry documenting a concept for which no standard term exists, the term entry will have to contain a "dummy" <tig>, which contains a empty term element of the form <term></term>.

Part: base tag set for terminological data
Member of classes: comp.terminology
DTD file: teite2n
Data description: May contain character data and phrase-level elements.
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage view
May contain: admin descrip tig [May include at any level: %m.terminologyInclusions]
Declaration:

```
<!ELEMENT termEntry     - O  ((%m.terminologyMisc)*, tig+)
                                          +(%m.terminologyInclusions)
                                                            >

<!ATTLIST termEntry          %a.global
          type               CDATA               #IMPLIED       >
```

Discussed in: 13.4.1 ('DTD Fragment for Nested Style') on p. 321; 13.3 ('Basic Structure of the Terminological Entry') on p. 316

text

<text> contains a single text of any kind, whether unitary or composite, for example a poem or drama, a collection of essays, a novel, a dictionary, or a corpus sample.

Attributes: [None: global and inherited attributes only.]
Example:

```
<text>
<front><docTitle>Autumn Haze</docTitle>
<body>
<l>Is it a dragonfly or a maple leaf
<l>That settles softly down upon the water?
</body>
</text>
```

The body of a text may be replaced by a group of nested texts, as in the following schematic:
Example:

```
<text>
 <front>
    <!-- front matter for whole text -->
 </front>
 <group>
   <!-- start of first group -->
  <text>
    <front>
      <!-- front matter for first text -->
    </front>
  </text>
  <text>
     <!-- second text -->
  </text>
 </group>
 <group>
  <!-- start of second group  -->
 </group>
</text>
```

Part: base tag set for common core features
Member of classes: declaring, inter
DTD file: teistr2
Occurs within: add admin camera caption case cell colloc corr country damage def desc descrip docEdition emph equiv etym figure figDesc foreign form fsDescr fDescr gen gram gramGrp group head hi hyph imprimatur item itype l lang lbl lem meeting mood note number orth otherForm p per pos pron q quote rdg ref region rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref TEI.2
May contain: back body front group [May include at any level: %m.globincl]
Declaration:

```
<!ELEMENT text          - -  (front?, (body | group), back?)
                                                +(%m.globincl;)
                                                              >

<!ATTLIST text                %a.global
                              %a.declaring                    >
```

Discussed in: 7 ('Default Text Structure') on p. 183; 13.4 ('Overall Structure of Terminological Documents') on p. 319; 23.1 ('Varieties of Composite Text') on p. 538

textClass

<textClass> (i.e. text classification) groups information which describes the nature or topic of a text in terms of a standard classification scheme, thesaurus, etc.

Attributes: [None: global and inherited attributes only.]
Example:

```
<textClass>
  <catRef target=TC1>
  <classCode source=DD12>001.9
```

```
    <keywords>
      <list><item>End of the world
      <item>History - philosophy
      </list>
  </textClass>
```

Part: auxiliary tag set for TEI headers
Member of classes: declarable
DTD file: teihdr2
Occurs within: profileDesc
May contain: catRef classCode keywords
Declaration:

```
<!ELEMENT textClass     - -  ((classCode | catRef | keywords)*
                              )                                  >

<!ATTLIST textClass           %a.global
                              %a.declarable                      >
```

Discussed in: 5.4.3 ('The Text Classification') on p. 111

textDesc

<textDesc> (i.e. text description) provides a description of a text in terms of its *situational parameters*.

Attributes: [None: global and inherited attributes only.]
Example:

```
<textDesc id=t1 n='Informal domestic conversation'>
  <channel mode=s>
  <constitution type=single>
  <derivation type=original>
  <domain type=domestic>
  <factuality type=mixed>
  <interaction type=complete active=plural passive=many>
  <preparedness type='spontaneous'>
  <purpose type=entertain degree=high>
  <purpose type=inform degree=medium>
</textDesc>
```

Part: auxiliary tag set for corpora and collections
Member of classes: declarable
DTD file: teicorp2
Data description:
Occurs within: catDesc profileDesc
May contain: channel constitution derivation domain factuality interaction preparedness purpose
Declaration:

```
<!ELEMENT textDesc      - O  (channel, constitution,
                              derivation, domain, factuality,
                              interaction, preparedness,
                              purpose+)                          >

<!ATTLIST textDesc            %a.global
                              %a.declarable                      >
```

Discussed in: 23.2.1 ('The Text Description') on p. 541

then

<then> separates the condition from the default in an <if>, or the antecedent and the consequent in a <cond> element.

Attributes: [None: global and inherited attributes only.]
This element is provided primarily to enhance the human readability of the feature-system declaration.
Part: auxiliary tag set for feature system declarations
Member of classes:
DTD file: teifsd2

Data description: Empty.
Occurs within: cond if
May contain: [none]
Declaration:

```
<!ELEMENT then          - O  EMPTY                          >

<!ATTLIST then                %a.global                     >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

tig

<tig> (i.e. term information group) within a <termEntry> element, contains information elements associated with a single term.

Attributes:

type classifies the <tig> using some typology, preferably the dictionary of data element types specified in ISO WD 12 620.
Data type: CDATA
Value: any string of characters; for serious terminological work, values should be taken from the dictionary of data element types specified in ISO WD 12 620.
Default: #IMPLIED

Example:

The <tig> element occurs only when the "nested" form of the terminology base is used. In the case of a term entry documenting a concept for which no standard term exists, the term entry will have to contain a "dummy" <tig>, which contains a empty term element of the form <term></term>.
Part: base tag set for terminological data
Member of classes:
DTD file: teite2n
Data description: May contain a <term> element, followed by zero or more <descrip> elements describing the term, followed in turn by zero or more <otherform> or <ofig> elements.
Occurs within: termEntry
May contain: admin descrip gram ofig term
Declaration:

```
<!ELEMENT tig           - O  ((%m.terminologyMisc)*, (term,
                             gram*), (%m.terminologyMisc)*,
                             ofig*)                        >

<!ATTLIST tig                 %a.global
          type                CDATA              #IMPLIED   >
```

Discussed in: 13.4.1 ('DTD Fragment for Nested Style') on p. 321; 13.2 ('Tags for Terminological Data') on p. 312

time

<time> contains a phrase defining a time of day in any format.

Attributes:

value gives the value of the time in a standard form.
Data type: CDATA
Value: Any string representing a time in standard format; recommended form is "hhmm", using the 24 hour clock.
Default: #IMPLIED
Example:

```
<time value=1600>four o' clock</time>
```

If the normalized form of time is not given using the twenty-four hour clock, this fact should be documented in the <stdVals> element in the TEI Header.

type indicates something about the type of temporal expression being tagged.
Data type: (AM | PM | 24HOUR | DESCRIPTIVE)
Sample values include:

am indicates a temporal expression made on the basis of a twelve-hour clock and referring to a time between midnight and noon.

pm indicates a temporal expression made on the basis of a twelve-hour clock and referring to a time between noon and midnight.

24hour indicates a temporal expression made on the basis of a twenty-four-hour clock.

descriptive indicates a temporal expression made in descriptive terms, e.g. "noon".

Default: #IMPLIED

zone indicates time zone or place name wherever this is necessary to evaluate a temporal expression.
Data type: CDATA
Value: contains a word or phrase such as GMT or 'Eastern Standard Time' which might be helpful in evaluating the temporal expression.
Default: #IMPLIED

Example:

```
As he sat smiling, the quarter struck &mdash <time value=1145>
the quarter to twelve</time>.
```

Part: additional tag set for common core features
Member of classes: data [and indirectly also:] phrase
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateline dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set setting settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT time          - -  (%phrase.seq;)                    >

<!ATTLIST time               %a.global
          value              CDATA               #IMPLIED
          type               (am | pm | 24hour | descriptive)
                                                 #IMPLIED
          zone               CDATA               #IMPLIED      >
```

Discussed in: 6.4.4 ('Dates and times') on p. 137

timeline

<timeline> (i.e. timeline) provides a set of ordered points in time which can be linked to elements of a spoken text to create a temporal alignment of that text.

Attributes:

origin designates the origin of the timeline, i.e. the time at which it begins.
Data type: IDREF
Value: must point to one of the <when> tags in its content.

Default: #REQUIRED
If the time of the origin is not known, it should be given an absolute value in descriptive terms, e.g. "sometime on the night of the murder" or "unknown".

unit specifies the unit of time corresponding to the interval value of the timeline or of its constituent points in time.
Data type: NMTOKEN
Value: a semi-closed list of recognized time units such as "millisecond", "second", "minute".
Default: #IMPLIED

interval specifies the numeric portion of a time interval
Data type: NUTOKEN
Value: -1 or any nonnegative number
Default: #IMPLIED
The value -1 indicates uncertainty about all the intervals in the timeline; 0 indicates that all the intervals are evenly spaced, but the size of the intervals is not known; positive values indicate evenly spaced values of the size specified. If individual points in time in the timeline are given different values for the interval attribute, those values locally override the value given in the timeline.

Example:

```
<timeline id=tl1 origin=w0 unit=centisecond>
  <when id=w0 absolute='sometime Monday morning before Christmas'
  <when id=w1 since=w0 interval=-1>
  <when id=w2 since=w1 interval=10>
  <when id=w3 since=w2 interval=20>
  <when id=w4 since=w3 interval=15>
  <when id=w5 since=w4 interval=25>
  <when id=w6 since=w5 interval=10>
  </timeline>
```

Occurs where??
Part: additional tag set for
Member of classes:
DTD file: teilink2.dtd
Data description: one or more points in time, one of which is its origin
Occurs within: [none]
May contain: when
Declaration:

```
<!ELEMENT timeline      - -  ((when)+)                          >

<!ATTLIST timeline           %a.global
          origin             IDREF               #REQUIRED
          unit               NMTOKEN             #IMPLIED
          interval           NUTOKEN             #IMPLIED       >
```

Discussed in: 14.5.2 ('Placing Synchronous Events in Time') on p. 366

timeRange

<timeRange> (i.e. time range) contains two times or another phrase indicating a time period.

Attributes:

from indicates the starting point of the time period in standard form.
Data type: CDATA
Value: a time of day; recommended form is to use the 24 hour clock.
Default: #IMPLIED
The value should conform to the standard form declared in the <stdVals> element in the TEI header.

to indicates the ending point of the time period in standard form.
Data type: CDATA
Value: a time of day; recommended form is to use the 24 hour clock.
Default: #IMPLIED

The value should conform to the standard form declared in the <stdVals> element in the TEI header.

exact indicates the precision to be attached to either or both times specified.

Data type: (TO | FROM | BOTH | NONE)

Sample values include:

to the to time is exact

from the from time is exact

both both times are exact

none both times are approximate or unspecified

Default: #IMPLIED

Example:

```
<timeRange from=1600 to=1620 exact=from>
     Just after teatime</timeRange>
```

time values should conform to the standard form declared in the <stdVals> element in the TEI header.

Part: additional tag set for common core features

Member of classes: data [and indirectly also:] phrase

DTD file: teicore2

Data description: May contain character data and phrase-level elements.

Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT timeRange     - -  (%phrase.seq;)                    >

<!ATTLIST timeRange           %a.global
          from                CDATA               #IMPLIED
          to                  CDATA               #IMPLIED
          exact               (to | from | both | none)
                                                  #IMPLIED      >
```

Discussed in: 6.4.4 ('Dates and times') on p. 137

<timeStruct> contains an internally structured representation for a time of day. timeStruct

Attributes:

zone indicates time zone or place name wherever this is necessary to evaluate a temporal expression.

Data type: CDATA

Value: contains a word or phrase such as GMT or "Eastern Standard Time" defining a time zone.

Default: #IMPLIED

Example:

```
<eg><![ CDATA [
<timeStruct value=1615 zone=GMT>
```

```
    <hour>four</hour>
    <minute>fifteen</minute>
  </timeStruct>
```

Part: additional tag set for common core features
Member of classes: data, temporalExpr [and indirectly also:] phrase
DTD file: teind2
Data description: May contain temporal components and character data only.
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange dateStruct def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange timeStruct title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA dateStruct day distance hour minute month occasion offset second timeStruct week year
Declaration:

```
<!ELEMENT timeStruct    - -  ((%m.temporalExpr; | #PCDATA)+)    >

<!ATTLIST timeStruct          %a.global
                              %a.temporalExpr
          zone                CDATA                #IMPLIED       >
```

Discussed in: 6.4.4 ('Dates and times') on p. 137

title

<title> contains the title of a work, whether article, book, journal, or series, including any alternative titles or subtitles.

Attributes:

level (i.e. bibliographic level (or class) of title) indicates whether this is the title of an article, book, journal, series, or unpublished material.
Data type: (A | M | J | S | U)
Sample values include:

- *a* analytic title (article, poem, or other item published as part of a larger item)
- *m* monographic title (book, collection, or other item published as a distinct item, including single volumes of multi-volume works)
- *j* journal title
- *s* series title
- *u* title of unpublished material (including theses and dissertations unless published by a commercial press)

Default: #IMPLIED
If the title appears directly enclosed within an <analytic> element, the level, if given, must be "a"; if it appears directly enclosed within a <monogr> element, level must be "m", "j", or "u"; when <title> is directly enclosed by <series>, level must be "s".

type (i.e. type of title) classifies the title according to some convenient typology.
Data type: CDATA
Sample values include:

- *main* main title
- *subordinate* subtitle, title of part
- *parallel* alternate title, often in another language, by which the work is also known
- *abbreviated* abbreviated form of title

Default: #IMPLIED

This attribute is provided for convenience in analysing titles and processing them according to their type; where such specialized processing is not necessary, there is no need for such analysis, and the entire title, including subtitles and any parallel titles, may be enclosed within a single <title> element.

Example:

```
<title>La vie mode d'emploi. Romans.</title>
```

Example:

```
<title>Synthese: an international journal for epistemology, methodology
and history of science</title>
```

Example:

```
<title>Information Technology and the Research Process: Proceedings
of a conference held at Cranfield Institute of Technology, UK, 18-21
July 1989</title>
```

Example:

```
<title>Hardy's Tess of the D'Urbervilles: a machine readable edition
```

Part: base tag set for common core features
Member of classes: hqphrase [and indirectly also:] phrase
DTD file: teicore2
Data description:
Occurs within: abbr activity actor add addrLine addName admin affiliation analytic author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned monogr mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense series seriesStmt set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart titleStmt tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT title         - O  (%paraContent)                    >

<!ATTLIST title              %a.global
          level              (a | m | j | s | u) #IMPLIED
          type               CDATA               #IMPLIED       >
```

Discussed in: 6.10.2.2 ('Authors, Titles, and Editors') on p. 168; 5.2.1 ('The Title Statement') on p. 82; 5.2.5 ('The Series Statement') on p. 88

titlePage

<titlePage> (i.e. title page) contains the title page of a text, appearing within the front or back matter.

Attributes:

type classifies the title page according to any convenient typology.
Data type: CDATA
Value: Any string, e.g. "full", "half", "Series", etc.
Default: #IMPLIED
This attribute allows the same element to be used for volume title pages, series title pages, etc., as well as for the "main" title page of a work.

Example:

```
<titlePage>
<docTitle>
<titlePart type=main>THOMAS OF
Reading.</>
<titlePart type=alt>
OR,
The sixe worthy yeomen
of the West.</>
</docTitle>
<docEdition>Now the fourth time corrected and enlarged</docEdition>
<byline>By T.D.</byline>
<ornament desc='emblem of TP'>
<s>TP
<s>Thou shalt labor till thou returne to duste
</ornament>
<docImprint>
Printed at <place>London</> for <name>T.P.</name>
<date>1612.</docImprint>
</titlePage>
```

Part: auxiliary tag set for common core features
Member of classes: front
DTD file: teifron2
Occurs within: back front
May contain: byline docAuthor docDate docEdition docImprint docTitle epigraph imprimatur titlePart
Declaration:

```
<!ELEMENT titlePage     - O  (%m.tpParts;)+                    >

<!ATTLIST titlePage          %a.global
          type               CDATA              #IMPLIED       >
```

Discussed in: 7.5 ('Title Pages') on p. 203

titlePart

<titlePart> (i.e. title part) contains a subsection or division of the title of a work, as indicated on a title page.

Attributes:

type specifies the role of this subdivision of the title.
Data type: CDATA
Sample values include:

main main title of the work
sub subtitle of the work
alt alternative title of the work
desc descriptive paraphrase of the work included in title

Default: MAIN

Example:

```
<docTitle>
  <titlePart type=main>THE
  FORTUNES
  AND
  MISFORTUNES
  Of the FAMOUS
  Moll Flanders, &c.
  <titlePart type=desc>Who was BORN in
  NEWGATE,
  And during a Life of continu'd Variety for
  Threescore Years, besides her Childhood, was
  Twelve Year a {Whore}, five times a {Wife} (wherof
  once to her own Brother) Twelve Year a {Thief,}
  Eight Year a Transported {Felon} in {Virginia}, at last
```

```
    grew {Rich}, liv'd {Honest}, and died a {Penitent}.
  </docTitle>
```

Part: base tag set for common core features
Member of classes: tpParts
DTD file: teifron2
Data description:
Occurs within: docTitle titlePage
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT titlePart     - O  (%paraContent;)                  >
<!ATTLIST titlePart          %a.global
          type               CDATA              main         >
```

Discussed in: 7.5 ('Title Pages') on p. 203

titleStmt

<titleStmt> (i.e. title statement) groups information about the title of a work and those responsible for its intellectual content.

Attributes: [None: global and inherited attributes only.]
Example:

```
<titleStmt>
    <title>Capgrave's Life of St. John Norbert:  a machine-readable
         transcription</title>
    <respStmt>
         <resp>compiled by</resp> <name>P.J. Lucas</name>
    </respStmt>
</titleStmt>
```

Part: auxiliary tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description:
Occurs within: biblFull fileDesc
May contain: author editor funder principal respStmt sponsor title
Declaration:

```
<!ELEMENT titleStmt     - O  ((title+, (author | editor |
                             sponsor | funder | principal |
                             respStmt)*))                  >
<!ATTLIST titleStmt          %a.global                     >
```

Discussed in: 5.2.1 ('The Title Statement') on p. 82; 5.2 ('The File Description') on p. 80

tns

<tns> (i.e. tense) indicates the grammatical tense associated with a given inflected form in a dictionary.

Attributes: [None: global and inherited attributes only.]
This element is synonymous with <gram type=tense>.
Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, morphInfo
DTD file: teidict2
Data description: May contain character data and phrase-level elements.
Occurs within: etym trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT tns           - O  (%paraContent;)                >
<!ATTLIST tns                %a.global
                             %a.dictionaries                >
```

Discussed in: 12.3.1 ('Information on Written and Spoken Forms') on p. 279

tr

<tr> (i.e. translation equivalent) contains a translation of the headword or an example.

Attributes: [None: global and inherited attributes only.]

This element is provided as a convenient means to distinguish translation equivalents from other methods of providing sense information, in the cases where such a distinction is desired. Where no such distinction is desired, translation equivalents may be tagged as <def> or <gloss> elements.

Part: base tag set for printed dictionaries

Member of classes: dictionaries, dictionaryParts

DTD file: teidict2

Data description: May contain character data and phrase-level elements.

Occurs within: etym trans

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref

Declaration:

```
<!ELEMENT tr            - O  (%paraContent;)                >
<!ATTLIST tr                 %a.global
                             %a.dictionaries                >
```

Discussed in: 12.3.3.2 ('Translation Equivalents') on p. 287

trailer

<trailer> (i.e. trailer) contains a closing title or footer appearing at the end of a division of a text.

Attributes: [None: global and inherited attributes only.]

Example:

```
<trailer>Explicit pars tertia</>
```

Part: base tag set for common core features

Member of classes: divbot

DTD file: teistr2

Occurs within: body castGroup div div0 div1 div2 div3 div4 div5 div6 div7 epilogue group lg listBibl performance prologue

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT trailer       - O  (%phrase.seq;)                 >
<!ATTLIST trailer            %a.global                      >
```

Discussed in: 7.2.4 ('Content of Textual Divisions') on p. 194; 7.2 ('Elements Common to All Divisions') on p. 190

trans

<trans> (i.e. translation information) contains translation text and related information (within an entry in a multilingual dictionary).

Attributes: [None: global and inherited attributes only.]

Part: base tag set for printed dictionaries

Member of classes: dictionaries, dictionaryParts, dictionaryTopLevel

DTD file: teidict2

Data description: May contain character data mixed with any other elements defined in the dictionary tag set.

Occurs within: entry etym hom re sense trans

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption case castList cit cl colloc corr date dateRange dateStruct def del distinct eg emph etym expan figure foreign form formula gap gen gi gloss gramGrp handShift hi hom hyph itype label lang lbl link list listBibl m measure mentioned mood move name note num number orig orth oRef oVar per phr pos pron ptr pRef pVar q quote re ref reg rs s seg sense sic sound soCalled stage stress subc superentry syll table tag tech term text time timeRange timeStruct title tns tr trans usg val view w xptr xr xref

Declaration:

```
<!ELEMENT trans         - O  (%paraContent |
                              %m.dictionaryParts)*              >

<!ATTLIST trans              %a.global
                              %a.dictionaries                   >
```

Discussed in: 12.3.3.2 ('Translation Equivalents') on p. 287

<tree> encodes a tree, which is made up of a root, internal nodes, leaves, and arcs from root to leaves. tree

Attributes:

label gives a label for a tree.
Data type: CDATA
Value: A character string.
Default: #IMPLIED

arity gives the maximum number of children of the root and internal nodes of the tree.
Data type: NUMBER
Value: A nonnegative integer.
Default: #IMPLIED

ord indicates whether or not the tree is ordered, or if it is partially ordered.
Data type: (Y | N | PARTIAL)
Sample values include:

Y indicates that all of the branching nodes of the tree are ordered.

partial indicates that some of the branching nodes of the tree are ordered and some are unordered.

N indicates that all of the branching nodes of the tree are unordered.

Default: Y

order gives the order of the tree, i.e., the number of its nodes.
Data type: NUMBER
Value: A nonnegative integer.
Default: #IMPLIED
The size of a tree is always one less than its order, hence there is no need for both a size and order attribute.

Example:

```
<tree n='ex2' ord=partial arity=2 order=13>
   <root  label='/'  id=div1   ord=Y             children='plu1 exp1'>
   <iNode label='+'  id=plu1   ord=N parent=div1 children='exp2 exp3'>
   <iNode label='**' id=exp1   ord=Y parent=div1 children='plu2 num2.3'>
   <iNode label='**' id=exp2   ord=Y parent=plu1 children='vara1 num2.1'>
   <iNode label='**' id=exp3   ord=Y parent=plu1 children='varb1 num2.2'>
   <iNode label='+'  id=plu2   ord=N parent=exp1 children='vara2 varb2'>
   <leaf  label='a'  id=vara1        parent=exp2>
   <leaf  label='2'  id=num2.1       parent=exp2>
   <leaf  label='b'  id=varb1        parent=exp3>
   <leaf  label='2'  id=num2.2       parent=exp3>
   <leaf  label='a'  id=vara2        parent=plu2>
   <leaf  label='b'  id=varb2        parent=plu2>
   <leaf  label='2'  id=num2.3       parent=exp1>
   </tree>
```

Part: additional tag set for graphs, networks, and trees

Member of classes: chunk [and indirectly also:] common
DTD file: teinet2
Data description: A root, and zero or more internal nodes and leaves, but if there is an internal node, there must also be at least one leaf.
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv forest item metDecl note performance prologue q quote remarks set sic stage view
May contain: iNode leaf root
Declaration:

```
<!ELEMENT tree          - -  ((leaf | iNode)*, root, (leaf |
                             iNode)*)                          >

<!ATTLIST tree               %a.global
          label              CDATA               #IMPLIED
          arity              NUMBER              #IMPLIED
          ord                (Y | N | partial)   Y
          order              NUMBER              #IMPLIED       >
```

Discussed in: 21.2 ('Trees') on p. 514

triangle

<triangle> (i.e. Underspecified embedding tree, so called because of its characteristic shape when drawn.) provides for an underspecified <eTree>, that is, an <eTree> with information left out.

Attributes:

label gives a label for an underspecified embedding tree.
Data type: CDATA
Value: A character string.
Default: #IMPLIED

value provides the value of a triangle, which is the SGML identifier of a feature structure or other analytic element.
Data type: IDREF
Value: A valid identifier of a feature structure or other analytic element.
Default: #IMPLIED

Example:

```
<triangle label='NP'><eLeaf label='the periscope'></triangle>
```

Part: additional tag set for graphs, networks, and trees
Member of classes:
DTD file: teinet2
Data description: An optional label or feature structure followed by zero or more embedding trees, triangles, or embedding leafs.
Occurs within: eTree forest triangle
May contain: eLeaf eTree triangle
Declaration:

```
<!ELEMENT triangle      - -  ((eTree | triangle | eLeaf)*)      >

<!ATTLIST triangle           %a.global
          label              CDATA               #IMPLIED
          value              IDREF               #IMPLIED       >
```

Discussed in: 21.3 ('Another Tree Notation') on p. 517

tsd

<tsd> (i.e. tag set documentation) contains a set of elements documenting an SGML tag set.

Attributes: [None: global and inherited attributes only.]
Example:

```
<tsd>
<tagDoc> ... </tagDoc>
<entDoc type=pe> ... </entDoc>
<tagDoc> ... </tagDoc>
</tsd>
```

This element is required.
Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: Contains any mixture of <tagDoc>, <entDoc>, and <classDoc> elements.
Occurs within: [none]
May contain: classDoc entDoc tagDoc
Declaration:

```
<!ELEMENT tsd           - O  ((tagDoc | entDoc | classDoc)+)    >
<!ATTLIST tsd                %a.global                          >
```

Discussed in: 27 ('Tag Set Documentation') on p. 601

<u> (i.e. utterance) a stretch of speech usually preceded and followed by silence or by a change of speaker. u

Attributes:

trans (i.e. transition) indicates the nature of the transition between this utterance and the previous one.
Data type: (SMOOTH | LATCHING | OVERLAP | PAUSE)
Sample values include:

smooth this utterance begins without unusual pause or rapidity.
latching this utterance begins with a markedly shorter pause than normal.
overlap this utterance begins before the previous one has finished.
pause this utterance begins after a noticeable pause.

Default: SMOOTH

who supplies an identifier for the speaker or group of speakers. Its value is the identifier of a <participant> or <participantGrp> element in the TEI header.
Data type: IDREF
Value: Must identify a participant or participant group within the TEI Header
Default: %INHERITED

Example:

```
<u who=A>if did you set</u>
<u who=B trans=latching>well Joe and I set it between us</u>
```

Although individual transcriptions may consistently use <u> elements for turns or other units, and although in most cases a <u> will be delimited by pause or change of speaker, <u> is not required to represent a turn or any communicative event, nor to be bounded by pauses or change of speaker. At a minimum, a <u> is some phonetic production by a given speaker.
Part: base tag set for spoken materials
Member of classes: comp.spoken, timed
DTD file: teispok2
Data description: prose and a mixture of speech elements
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage u view
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph event expan foreign formula gap gi gloss handShift hi kinesic lang link m measure mentioned name num orig oRef oVar pause phr ptr pRef pVar ref reg rs s seg shift sic soCalled tag term time timeRange timeStruct title u val vocal w writing xptr xref
Declaration:

```
<!ELEMENT u             - -  ((%phrase | %m.comp.spoken)+)      >
<!ATTLIST u                  %a.global
                             %a.timed
          trans              (smooth | latching | overlap |
                             pause)              smooth
          who                IDREF               %INHERITED     >
```

Discussed in: 11.2.7 ('Formal Definition') on p. 259; 11.2.1 ('Utterances') on p. 254; 11.2 ('Elements Unique to Spoken Texts') on p. 253

uncertain

<uncertain> (i.e. Uncertain value) provides uncertainty value for a feature.

Attributes: [None: global and inherited attributes only.]
Example:

```
<f name=gender><uncertain>
```

Part: additional tag set for feature structures
Member of classes: singleVal [and indirectly also:] featureVal
DTD file: teifs2
Data description: Empty element.
Occurs within: f fvLib if vAlt vDefault vRange
May contain: [none]
Declaration:

```
<!ELEMENT uncertain     - O  EMPTY                              >
<!ATTLIST uncertain          %a.global                          >
```

Discussed in: 16.8 ('Boolean, Default and Uncertain Values') on p. 417

unclear

<unclear> contains a word, phrase, or passage which cannot be transcribed with certainty because it is illegible or inaudible in the source.

Attributes:

reason indicates why the material is hard to transcribe.
Data type: CDATA
Value: any phrase describing the difficulty, e.g. "faded", "ambient noise", "passing truck", "ill formed", "eccentric ductus".
Default: #IMPLIED

resp indicates the individual responsible for the transcription of the letter, word or passage contained with the <unclear> element.
Data type: CDATA
Value: must be one of the identifiers declared in the document header, associated with a person asserted as responsible for some aspect of the text's creation, transcription, editing or encoding (see chapter 17 ('Certainty and Responsibility') on p. 435).
Default: %INHERITED

cert (i.e. certainty) signifies the degree of certainty ascribed to the transcription of the text contained within the <unclear> element.
Data type: CDATA
Default: #IMPLIED

hand Where the difficulty in transcription arises from action (partial deletion, etc.) assignable to an identifiable hand, signifies the hand responsible for the action.
Data type: IDREF
Value: must be one of the hand identifiers declared in the document header (see section 18.2.1 ('Document Hands') on p. 456).
Default: %INHERITED

agent Where the difficulty in transcription arises from an identifiable cause, signifies the causative agent.
Data type: CDATA
Value: any prose description of the agent.
Default: #IMPLIED

The same element is used for all cases of uncertainty in the transcription of element content, whether for written or spoken material. For other aspects of certainty, uncertainty, and reliability of tagging and transcription, see chapter 17 ('Certainty and Responsibility') on p. 435.

The <damage>, <omit>, <del>, <unclear> and <supplied> elements may be closely allied in use. See section 18.2.4 ('The Use of the Gap, Del, Damage, Unclear and Supplied Tags in Combination') on p. 462 for discussion of which element is appropriate for which circumstance.

Part: base tag set for common core features
Member of classes:
DTD file: teicore2
Data description: May contain character data and phrase-level elements.
Occurs within: [none]
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT unclear       - O  (%paraContent;)                    >

<!ATTLIST unclear            %a.global
          reason             CDATA               #IMPLIED
          resp               CDATA               %INHERITED

          cert               CDATA               #IMPLIED
          hand               IDREF               %INHERITED

          agent              CDATA               #IMPLIED       >
```

Discussed in: 18.2.3 ('Damage, Illegibility, and Supplied Text') on p. 460; 6.5.3 ('Additions, Deletions and Omissions') on p. 144

<usg> (i.e. usage) contains usage information in a dictionary entry. usg

Attributes:

type classifies the usage information using any convenient typology.
Data type: CDATA
Sample values include:

geo geographic area
time temporal, historical era ("archaic", "old", etc.)
dom domain
reg register
style style (figurative, literal, etc.)
plev preference level ("chiefly", "usually", etc.)
lang lang (language for foreign words, spellings pronunciations, etc.)
gram grammatical usage
syn synonym given to show use
hyper hypernym given to show usage
colloc collocation given to show usage
comp typical complement
obj typical object
subj typical subject
verb typical verb
hint unclassifiable piece of information to guide sense choice

Default: #IMPLIED

Part: base tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, dictionaryTopLevel, formInfo, gramInfo
DTD file: teidict2
Data description: May contain character data and phrase-level elements.
Occurs within: entry etym form gramGrp hom re sense trans xr
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT usg           - O  (%paraContent;)                    >
<!ATTLIST usg                %a.global
                             %a.dictionaries
          type               CDATA                #IMPLIED     >
```

Discussed in: 12.3.5.2 ('Usage Information and Other Labels') on p. 291

val

<val> (i.e. value) contains a single attribute value.

Attributes: [None: global and inherited attributes only.]
Example:

```
<val>unknown</val>
```

Part: auxiliary tag set for tag set declarations
Member of classes: sgmlKeywords [and indirectly also:] phrase
DTD file: teitsd2
Data description:
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc valList view wit witness witDetail writing xr xref
May contain: #PCDATA
Declaration:

```
<!ELEMENT val           - O  (#PCDATA)                          >
<!ATTLIST val                %a.global                          >
```

Discussed in: 27 ('Tag Set Documentation') on p. 601; 27.1.1 ('The AttList Documentation Element') on p. 605

valDesc

<valDesc> (i.e. value description) specifies any semantic or syntactic constraint on the value that an attribute may take, additional to the information carried by the <dataType> element.

Attributes: [None: global and inherited attributes only.]
Example:

```
<valDesc>must point to another <gi>align</gi>
   element logically preceding this
   one.</valDesc>
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description: contains any sequence of phrase level elements or pcdata
Occurs within: attDef
May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref
Declaration:

```
<!ELEMENT valDesc       - O  (%phrase.seq)                      >
<!ATTLIST valDesc            %a.global                          >
```

Discussed in: 27.1.1 ('The AttList Documentation Element') on p. 605

valList

<valList> (i.e. value list) contains a list of value and description pairs for an attribute.

Attributes:

type specifies the extensibility of the list of attribute values specified.
Data type: (CLOSED | SEMI | OPEN)
Sample values include:

closed only the values specified are permitted.

semi all the values specified should be supported, but other values are legal and software should have appropriate fallback processing for them.

open the values specified are sample values only.

Default: OPEN

Example:

```
<valList>
 <val>req<desc>required
 <val>mwa<desc>mandatory when applicable
 <val>rec<desc>recommended
 <val>rwa<desc>recommended when applicable
 <val>opt<desc>optional
</valList>
```

Part: auxiliary tag set for tag set declarations
Member of classes:
DTD file: teitsd2
Data description:
Occurs within: attDef
May contain: desc val
Declaration:

```
<!ELEMENT valList       - -  ((val, desc)*)                    >

<!ATTLIST valList            %a.global
          type               (closed | semi | open)
                                                 open          >
```

Discussed in: 27.1.1 ('The AttList Documentation Element') on p. 605

<vAlt> (i.e. Value alternation) provides alternative (disjunctive) values for a feature. vAlt

Attributes:

mutExcl indicates whether values are mutually exclusive.
Data type: (Y | N)
Sample values include:

Y indicates that the values are mutually exclusive.

N indicates that the values are not mutually exclusive.

Default: #IMPLIED

Example:

```
<vAlt mutExcl=Y><sym value=masculine><sym value=neuter></vAlt>
```

Part: additional tag set for feature structures
Member of classes: complexVal [and indirectly also:] featureVal
DTD file: teifs2
Data description: Two or more feature values.
Occurs within: f fsLib fvLib if vAlt vDefault vRange
May contain: any dft fs minus msr nbr none null plus rate str sym uncertain vAlt
Declaration:

```
<!ELEMENT vAlt          - -  ((plus | minus | any | none | dft
                              | uncertain | null | sym | nbr |
                              msr | rate | str | vAlt | fs),
                              (plus | minus | any | none | dft |
                              uncertain | null | sym | nbr | msr
                              | rate | str | vAlt | fs)+)       >
```

```
<!ATTLIST vAlt              %a.global
          mutExcl           (Y | N)             #IMPLIED       >
```

Discussed in: 16.7 ('Alternative Features and Feature Values') on p. 413

variantEncoding

<variantEncoding> declares the method used to encode text-critical variants.

Attributes:

method indicates which method is used to encode the apparatus of variants.

Data type: (LOCATION-REFERENCED | DOUBLE-END-POINT | PARALLEL-SEGMENTATION)

Sample values include:

location-referenced apparatus uses line numbers or other canonical reference scheme referenced in a base text.

double-end-point apparatus indicates the precise locations of the beginning and ending of each lemma relative to a base text.

parallel-segmentation alternate readings of a passage are given in parallel in the text; no notion of a base text is necessary.

Default: #REQUIRED

The value "parallel-segmentation" requires in-line encoding of the apparatus.

location indicates whether the apparatus appears within the running text or external to it.

Data type: (INTERNAL | EXTERNAL)

Sample values include:

internal apparatus appears within the running text.

external apparatus appears outside the base text.

Default: #REQUIRED

The value "external" is inconsistent with the parallel-segmentation method of apparatus markup.

Example:

```
<variant.encoding method='location-referenced'
                  location='external'>
```

Part: base tag set for TEI headers
Member of classes:
DTD file: teihdr2
Data description: Empty.
Occurs within: [none]
May contain: [none]
Declaration:

```
<!ELEMENT variantEncoding
                        - O  EMPTY                          >

<!ATTLIST variantEncoding   %a.global
          method            (location-referenced |
                             double-end-point |
                             parallel-segmentation)
                                                #REQUIRED
          location          (internal | external)
                                                #REQUIRED     >
```

Discussed in: 19.1.1 ('The Apparatus Entry') on p. 469

vDefault

<vDefault> (i.e. value default) declares the default value to be supplied when a feature structure does not contain an instance of <f> for this name; if unconditional, it is specified as one (or, depending on the value of the org attribute of the enclosing <fDecl>) more <fs> elements or primitive values; if conditional, it is specified as one or more <if> elements; if no default is specified, or no condition matches, the value <none> is assumed.

Attributes: [None: global and inherited attributes only.]
Part: auxiliary tag set for feature system declarations
Member of classes:

DTD file: teifsd2
Data description: May contain a legal feature value, or a series of <if> elements.
Occurs within: fDecl
May contain: alt any dft fs if minus msr nbr none null plus rate str sym uncertain vAlt
Declaration:

```
<!ELEMENT vDefault      - -  ((%m.featureVal)+ | if+)          >
<!ATTLIST vDefault           %a.global                         >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

<%version> (i.e. version name) serves as the root element for a concurrent markup stream which will be used to mark page and line numbers of a reference edition of the text. %version

Attributes: [None: global and inherited attributes only.]
Example:

```
<!DOCTYPE TEI.2 system 'tei2.dtd' >
<!DOCTYPE La    system 'teip2.dtd' [
       <!ENTITY % version 'La'>
]>
<(TEI.2)TEI.2><(La)La>
<(TEI.2)TEI.Header> ...
</(TEI.2)TEI.Header>
<(TEI.2)text>
<(La)page n='[1]'>
  ...
<(La)page n='4'>
  ...
<(La)page n='5'>
  ...
</(TEI.2)text>
</(La)La>
</(TEI.2)TEI.2>
```

The version name should be short (one or two characters), as it must be prefixed to all page and line tags in the concurrent markup stream.
Part: base tag set for
Member of classes:
DTD file: teipl2
Data description: May contain character data, <vol>, and <page> elements.
Occurs within: [none]
May contain: #PCDATA page vol
Declaration:

```
<!ELEMENT %version      - -  (#PCDATA | page | vol)*           >
<!ATTLIST %version           %a.global                         >
```

Discussed in: 6.9 ('Reference Systems') on p. 155

<view> describes the visual context of some part of a screen play in terms of what the spectator sees, generally independent of any dialogue. view

Attributes: [None: global and inherited attributes only.]
Example:

```
<view><name>Max</name> joins his daughter
at the window. <hi>Rain</hi> sprays his
face-- </view>
<view><camera>Max's POV</camera> He sees occasional
windows open, and just across from his apartment
house, a <hi>man</hi> opens the front door of
a brownstone--</view>
```

Example:

```
<div type=shot>
  <view>BBC World symbol</view>
  <sp who=MP><speaker>Voice Over</speaker>
     <p>Monty Python's Flying Circus tonight comes to you live
     from the Grillomat Snack Bar, Paignton.
</div>
<div type=shot>
  <view>Interior of a nasty snack bar. Customers around, preferably
  real people. Linkman sitting at one of the plastic tables.</view>
  <sp who=JC><speaker>Linkman</speaker>
  <p>Hello to you live from the Grillomat Snack Bar.
  <!-- ... -->
</div>
```

A view is a particular form of stage direction.

Part: base tag set for performance texts

Member of classes: stageDirection [and indirectly also:] comp.drama, inter

DTD file: teidram2

Data description: text and crystals

Occurs within: add admin argument body camera caption case castList cell colloc corr country damage def desc descrip div div0 div1 div2 div3 div4 div5 div6 div7 docEdition emph epigraph epilogue equiv etym figDesc foreign form fsDescr fDescr gen gram gramGrp head hi hyph imprimatur item itype l lang lbl lem meeting metDecl mood note number orth otherForm p per performance pos prologue pron q quote rdg ref region remarks rendition seg set sic sound stage stress subc supplied syll tagUsage tech title titlePart tns tr trans unclear usg view witness witDetail writing xr xref

May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph entry entryFree event expan eTree figure foreign formula gap gi gloss graph handShift hi kinesic l label lang lg lg1 link list listBibl m measure mentioned move name note num orig oRef oVar p pause phr ptr pRef pVar q quote ref reg rs s seg shift sic sound soCalled sp stage superentry table tag tech term termEntry text time timeRange timeStruct title tree u val view vocal w writing xptr xref TEI...end

Declaration:

```
<!ELEMENT view          - O  (%specialPara)                    >

<!ATTLIST view                %a.global                        >
```

Discussed in: 10.3.1 ('Technical Information') on p. 248; 10.3 ('Other Types of Performance Text') on p. 246

vocal

<vocal> (i.e. Vocalized semi-lexical) any vocalized but not necessarily lexical phenomenon, for example voiced pauses, non-lexical backchannels, etc.

Attributes:

who supplies an identifier for the vocalist(s). Its value is the identifier of a <participant> or <participant.grp> element in the TEI header.
Data type: IDREF
Value: Must identify a participant or participant group within the TEI Header
Default: %INHERITED

iterated (i.e. iterated) indicates whether or not the phenomenon is repeated.
Data type: (Y | N | U)
Sample values include:

- *y* the phenomenon is repeated.
- *n* the phenomenon is atomic.
- *u* unknown or unmarked.

Default: N

desc (i.e. description) supplies a conventional representation for the phenomenon.
Data type: CDATA
Value: a description or representation of the phenomenon chosen from a semi-closed list
Default: #IMPLIED

Example:

```
<vocal dur=12 desc=whistles>
```

Part: base tag set for spoken materials
Member of classes: comp.spoken, timed
DTD file: teispok2
Data description: empty
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage u view
May contain: [none]
Declaration:

```
<!ELEMENT vocal          - O  EMPTY                              >

<!ATTLIST vocal               %a.global
                              %a.timed
          who                 IDREF               %INHERITED

          iterated            (y | n | u)         n
          desc                CDATA               #IMPLIED       >
```

Discussed in: 11.2.7 ('Formal Definition') on p. 259; 11.2.3 ('Vocal, Kinesic, Event') on p. 256; 11.2 ('Elements Unique to Spoken Texts') on p. 253

vol

<vol> (i.e. volume) marks the individual volumes of a reference edition.

Attributes: [None: global and inherited attributes only.]
Example:

```
<(La)vol  n=2>  ... [text of edition La, volume 2]
<(La)page n=32> ... [text of edition La, vol. 2, page 32]
<(La)page n=32> ... [text of edition La, vol. 2, page 33]
<(La)page n=32> ... [text of edition La, vol. 2, page 34]
<(La)page n=32> ... [text of edition La, vol. 2, page 35]
```

The <vol> may be used if the reference edition has more than one volume; otherwise it need not be used. Any data contained within a <vol> element but not within a <page> element is assumed not to appear in the edition from which the reference scheme derives.
Part: base tag set for
Member of classes:
DTD file: teipl2
Data description: May contain character data and <page> elements.
Occurs within: %version
May contain: #PCDATA page
Declaration:

```
<!ELEMENT vol            - -  (#PCDATA | page)*              >

<!ATTLIST vol                 %a.global                      >
```

Discussed in: 31.6 ('Concurrent Markup for Pages and Lines') on p. 637

vRange

<vRange> (i.e. value range) defines the range of allowed values for a feature, in the form of an <fs>, <vAlt>, or primitive value; for the value of an <f> to be valid, it must be *subsumed* by the specified range; if the <f> contains multiple values (as sanctioned by the org attribute), then each value must be subsumed by the <vRange>.

Attributes: [None: global and inherited attributes only.]
Part: auxiliary tag set for feature system declarations
Member of classes:
DTD file: teifsd2
Data description: May contain any legal feature-value specification.
Occurs within: fDecl
May contain: alt any dft fs minus msr nbr none null plus rate str sym uncertain vAlt
Declaration:

```
<!ELEMENT vRange         - O  (%m.featureVal)                >
```

```
<!ATTLIST vRange              %a.global                          >
```

Discussed in: 26 ('Feature System Declaration') on p. 589

w

<w> (i.e. word) represents a grammatical (not necessarily orthographic) word.

Attributes:

lemma identifies the word's lemma (dictionary entry form).

Data type: CDATA

Value: a string of characters representing the spelling of the word's dictionary entry form.

Default: #IMPLIED

Example:

```
<w type=verb lemma=hit>hitt<m type=suffix>ing</m></w>
```

Part: additional tag set for

Member of classes: seg [and indirectly also:] phrase

DTD file: teiana2

Data description: May contain character data and <seg>, <w>, <m> and <c> elements.

Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view w wit witness witDetail writing xr xref

May contain: #PCDATA c m seg w

Declaration:

```
<!ELEMENT w                   - -  ((#PCDATA | seg | w | m | c)*)     >
<!ATTLIST w                        %a.global
                                   %a.seg
          lemma                    CDATA                 #IMPLIED     >
```

Discussed in: 15.1 ('Linguistic Segment Categories') on p. 382

week

<week> (i.e. week) the week component of a structured date.

Attributes: [None: global and inherited attributes only.]

Example:

```
<title>Le Vieux Cordelier: Journals redige par Camille Desmoulins</title>,
<dateStruct type=Revolutionary value='03-02-1794'>
<day type=name>Quintidi</day>
<month>Pluviose</month>
<week value='2'>2e decade</week>,
<year>l'an 2 de la Republique Indivicible</year>
</dateStruct>
```

Part: additional tag set for names and dates

Member of classes: temporalExpr

DTD file: teind2

Occurs within: dateStruct timeStruct

May contain: #PCDATA

Declaration:

```
<!ELEMENT week                - -  (#PCDATA)                         >
<!ATTLIST week                     %a.global
                                   %a.temporalExpr                   >
```

Discussed in: 20.4 ('Dates and Time') on p. 499

<when> (i.e. when) indicates a point in time either relative to other elements in the same <timeline> tag, or absolutely. when

Attributes:

absolute supplies an absolute value for the time.
Data type: CDATA
Value: Times may be given in standard form, as specified in the Encoding Declarations section of the header.
Default: #IMPLIED
Required for the element designated as the value of the origin attribute in the <timeline> tag.

unit specifies the unit of time corresponding to the interval value.
Data type: NMTOKEN
Value: a semi-closed list of recognized time units such as "millisecond", "second", "minute"
Default: %INHERITED;

interval specifies the numeric portion of a time interval
Data type: NUTOKEN
Value: -1 or any positive number
Default: %INHERITED;
The value -1 indicates uncertainty about the interval.

since identifies the reference point for determining the time of the current <when> element, which is obtained by adding the interval to the time of the reference point.
Data type: IDREF
Value: Should point to another <when> element in the same <timeline>.
Default: #IMPLIED
If this attribute is omitted, and the absolute attribute is not specified, then the reference point is understood to be the origin of the enclosing <timeline> tag.

id supplies an identifier, unique to the document, for each <when> element.
Data type: ID
Value: any valid name
Default: #REQUIRED

Example:

```
<when id=w3 since=w2 interval=20>
```

Part: additional tag set for
Member of classes:
DTD file: teilink2.dtd
Data description: empty
Occurs within: timeline
May contain: [none]
Declaration:

```
<!ELEMENT when          - O  EMPTY                              >

<!ATTLIST when               %a.analysis
                             %a.linking
                             %a.terminology
          n                  CDATA               #IMPLIED
          lang               IDREF               %INHERITED

          rend               CDATA               #IMPLIED
          absolute           CDATA               #IMPLIED
          unit               NMTOKEN             %INHERITED;
          interval           NUTOKEN             %INHERITED;
          since              IDREF               #IMPLIED
          id                 ID                  #REQUIRED      >
```

Discussed in: 14.5.2 ('Placing Synchronous Events in Time') on p. 366

wit

<wit> contains a list of one or more sigla of witnesses attesting a given reading, in a textual variation.

Attributes: [None: global and inherited attributes only.]

Example:

```
<rdg wit='El Hg'>Experience</> <wit>El Hg</>
```

This element duplicates the information present in the wit attribute of the reading; it may be used to record the exact form of the sigla given in the source edition, when that is of interest.

Part: additional tag set for text criticism

Member of classes:

DTD file: teitc2

Data description: May contain character data and phrase-level elements.

Occurs within: app rdgGrp

May contain: #PCDATA abbr add address anchor att c caesura cl corr date dateRange dateStruct del distinct emph expan foreign formula gap gi gloss handShift hi lang link m measure mentioned name num orig oRef oVar phr ptr pRef pVar ref reg rs s seg sic soCalled tag term time timeRange timeStruct title val w xptr xref

Declaration:

```
<!ELEMENT wit           - O  (%phrase.seq;)                  >

<!ATTLIST wit                %a.global                       >
```

Discussed in: 19.1.4 ('Witness Information') on p. 475

witDetail

<witDetail> gives further information about a particular witness, or witnesses, to a particular reading.

Attributes:

target indicates the identifier for the reading, or readings, to which the witness detail refers.
Data type: IDREFS
Value: the identifier of the reading or readings.
Default: #REQUIRED

wit indicates the sigil or sigla for the witnesses to which the detail refers.
Data type: CDATA
Value: the identifier of the sigil or sigla.
Default: #REQUIRED
In local encoding schemes, the value of the wit attribute can be enforced as IDREFS, such that only witnesses referred to in a <witList> element may be the subject of a <witDetail>.

type describes the type of information given about the witness.
Data type: CDATA
Value: Values can be taken from any convenient typology of annotation suitable to the work in hand; e.g. letter form, ornament, ...
Default: #IMPLIED

place indicates where the note appears in the source text.
Data type: CDATA
Value: As for the <note> element.
Default: 'APPARATUS'
For pages with multiple apparatus, values such as *app1* and *app2* can be used.
The place attribute can be used to indicate to text formatting software where a note should be printed. If the locations indicated do not agree with those in the copy text, that fact should be indicated in the TEI header.

The <witDetail> element should be regarded as a specialized type of <note> element; it is synonymous with <note type='witness detail'>. It differs from the general purpose <note> in the omission of attributes seldom applicable to notes within critical apparatus, and in the provision of the wit attribute, which permits an application to extract all annotation concerning a particular witness or witnesses from the apparatus.

Part: additional tag set for text criticism
Member of classes:
DTD file: teitc2
Data description:
Occurs within: [none]
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT witDetail     - O  (%paraContent)                    >

<!ATTLIST witDetail          %a.global
          target             IDREFS              #REQUIRED
          wit                CDATA               #REQUIRED
          type               CDATA               #IMPLIED
          place              CDATA               'apparatus'    >
```

Discussed in: 19.1 ('The Apparatus Entry, Readings, and Witnesses') on p. 469

<witEnd> indicates the end, or suspension, of the text of a fragmentary witness. witEnd

Attributes: [None: global and inherited attributes only.]
(optional)
Part: additional tag set for text criticism
Member of classes: fragmentary
DTD file: teitc2
Data description: Empty.
Occurs within: [none]
May contain: [none]
Declaration:

```
<!ELEMENT witEnd        - O  EMPTY                             >

<!ATTLIST witEnd             %a.global
                             %a.fragmentary                    >
```

Discussed in: 19.1.5 ('Fragmentary Witnesses') on p. 479

<witList> (i.e. witness list) contains a list of all the witnesses referred to in <wit> elements or wit attributes within the critical apparatus. witList

Attributes: [None: global and inherited attributes only.]
The provision of a <witList> element simplifies the automatic processing of the apparatus, e.g. the reconstruction of the readings for all witnesses from an exhaustive apparatus.
Situations commonly arise where there are many more or less fragmentary witnesses, such that there may be quite distinct groups of witnesses for different parts of a text or collection of texts. One may treat this with distinct <witList> elements for each different part. Alternatively, one may have a single <witList> element at the beginning of the file listing all the witnesses, partial and complete, for the text, with the attestation of fragmentary witnesses indicated within the apparatus by use of the <witStart> and <witEnd> elements described in section 19.1.5 ('Fragmentary Witnesses') on p. 479.
Part: additional tag set for text criticism
Member of classes:
DTD file: teitc2
Data description: May contain a series of <witness> elements.
Occurs within: [none]
May contain: witness
Declaration:

```
<!ELEMENT witList       - O  (witness+)                        >

<!ATTLIST witList            %a.global                         >
```

Discussed in: 19.1 ('The Apparatus Entry, Readings, and Witnesses') on p. 469

witness

<witness> contains either a description of a single witness referred to within the critical apparatus, or a list of witnesses which is to be referred to by a single sigil.

Attributes:

sigil indicates the sigil for one witness or for one group of witnesses to which readings are assigned in a critical apparatus.
Data type: CDATA
Value: the identifier to be used for this witness or witness group in the wit attribute of readings in the apparatus.
Default: #REQUIRED
In local encoding schemes, the value of the id attribute can be used as the sigil, and the declared value of the wit attribute may be changed to IDREF, so as to ensure that only witnesses referred to in a <witness> element contained within a <witList> may occur in the value of any wit attribute on a reading element within an apparatus.

included indicates which other witnesses are included in a witness group.
Data type: CDATA
Value: a blank-delimited list of sigla.
Default: ''
Example:

```
<witness sigil='B'>
<witness sigil='k'>
<witness sigil='I'>
<witness sigil='C*' included='B k I'>
```

The included attribute applies only in the case of witness groups; if no value is given, the <witness> element is held to refer to a single witness rather than a group.

The content of the <witness> element may give bibliographic information about the witness or witness group, or it may be empty.

Part: additional tag set for text criticism
Member of classes:
DTD file: teitc2
Occurs within: witList
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT witness       - O  (%paraContent)                    >

<!ATTLIST witness            %a.global
          sigil              CDATA              #REQUIRED
          included           CDATA              ''             >
```

Discussed in: 19.1 ('The Apparatus Entry, Readings, and Witnesses') on p. 469

witStart

<witStart> indicates the beginning, or resumption, of the text of a fragmentary witness.

Attributes: [None: global and inherited attributes only.]
(optional)
Part: additional tag set for text criticism
Member of classes: fragmentary
DTD file: teitc2
Data description: Empty.
Occurs within: [none]
May contain: [none]
Declaration:

```
<!ELEMENT witStart      - O  EMPTY                             >
```

```
<!ATTLIST witStart          %a.global
                            %a.fragmentary                  >
```

Discussed in: 19.1.5 ('Fragmentary Witnesses') on p. 479

<writing> (i.e. Writing) a passage of written text revealed to participants in the course of a spoken text. writing

Attributes:

who (i.e. who) supplies an identifier for the participant who reveals or creates the writing, if any. Its value is the identifier of a <participant> or <participant.grp> element in the TEI header.
Data type: IDREF
Value: Must identify a participant or participant group within the TEI Header
Default: %INHERITED

type (i.e. Type) categorizes the kind of writing in some way, for example as a subtitle, noticeboard etc.
Data type: CDATA
Value: Open list
Default: #IMPLIED

script (i.e. Script pointer) points to a bibliographic citation in the header giving a full description of the source or script of the writing.
Data type: IDREF
Value: Must be a valid identifier for a <script.decl> element in the TEI header
Default: #IMPLIED

gradual (i.e. gradual) indicates whether the writing is revealed all at once or gradually.
Data type: (Y | N | U)
Sample values include:

- *y* the writing is revealed gradually.
- *n* the writing is revealed all at once.
- *u* unknown or unmarked.

Default: #IMPLIED

The <writing> element will usually be short and most simply transcribed as a character string; the content model also allows a sequence of paragraphs and paragraph-level elements, in case the writing has enough internal structure to warrant such markup. In either case the usual phrase-level tags for written text are available.

Part: base tag set for spoken materials
Member of classes: comp.spoken
DTD file: teispok2
Occurs within: add argument body castList corr div div0 div1 div2 div3 div4 div5 div6 div7 epigraph epilogue equiv item metDecl note performance prologue q quote remarks set sic stage u view
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT writing       - -  (%paraContent;)                  >
<!ATTLIST writing           %a.global
          who               IDREF               %INHERITED

          type              CDATA               #IMPLIED
          script            IDREF               #IMPLIED
          gradual           (y | n | u)         #IMPLIED      >
```

Discussed in: 11.2.7 ('Formal Definition') on p. 259; 11.2.4 ('Writing') on p. 257; 11.2 ('Elements Unique to Spoken Texts') on p. 253

<writingSystemDeclaration> declares the coded character set, transliteration scheme, or entity set used to transcribe a given writing system of a given language. writingSystemDeclaration

Attributes:

name gives a formal name for the writing system declaration
Data type: CDATA
Value: any string of characters
Default: #REQUIRED
Example:

```
<writingSystemDeclaration
    name='-//TEI P2: 1993//WSD ISO 646 IRV//en'
    date='1993-06-30'>
```

date gives the date on which the writing system declaration was last revised.
Data type: %ISO-DATE
Value: A date in valid ISO format.
Default: #REQUIRED

Example:

```
<writingSystemDeclaration id=eng lang=eng
  name='-//TEI P2: 1993//WSD Modern English//en'
  date='1993-05-25'>
  <language>Modern English</>
  <direction lines=TB chars=LR>
  <base name='ANSI X3.4' authority='national'>
  <exceptions></>
</writingSystemDeclaration>
```

Part: auxiliary tag set for writing system declarations
Member of classes:
DTD file: teiwsd2
Data description: May contain only a prescribed series of sub-elements.
Occurs within: [none]
May contain: characters direction language note script
Declaration:

```
<!ELEMENT writingSystemDeclaration
                          - -  (language, script, direction*,
                               characters, note*)                >
<!ATTLIST writingSystemDeclaration
                    %a.global
          name                 CDATA                #REQUIRED
          date                 %ISO-date            #REQUIRED     >
```

Discussed in: 25.1 ('Overall Structure of Writing System Declaration') on p. 571

xptr

<xptr> (i.e. extended pointer) defines a pointer to another location in the current document or an external document.

Attributes: [None: global and inherited attributes only.]
Part: additional tag set for
Member of classes: loc, terminologyInclusions, xPointer [and indirectly also:] phrase, pointer
DTD file: teilink2.dtd
Occurs within: abbr activity actor add addrLine addName admin affiliation altGrp author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cit cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype joinGrp l label lang language langKnown lbl lem linkGrp locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term termEntry time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref

May contain: [none]
Declaration:

```
<!ELEMENT xptr          - O  EMPTY                          >

<!ATTLIST xptr               %a.global
                             %a.xPointer                    >
```

Discussed in: 14.2 ('Extended Pointers') on p. 340

<xr> (i.e. cross-reference phrase) contains a phrase, sentence, or icon referring the reader to some other location in this or another text. xr

Attributes:

type indicates the type of cross reference, using any convenient typology.
Data type: CDATA
Sample values include:

syn cross reference for synonym information
etym etymological information
cf related or similar term
illus illustration of an object

Default: #IMPLIED
Example:

```
<xr type=cf>Compare <ref>madrigal (sense 1)</></xr>
<xr type=illus>see the illus. at <ref>tool</></xr>
```

Example:

```
<entry>
  <form><orth>lavage</>
  <!-- ... -->
  <etym>[Fr. < <mentioned>laver</>; L. <mentioned>lavare</>,
        to wash; <xr>see <ref>lather</ref></xr>].</etym>
  <!-- ... -->
</entry>
```

Example:

```
<entry>
  <form><orth>lawful</>
  <!-- ... -->
  <xr type='syn'>SYN. see <ref>legal</ref></xr>
</entry>
```

This element encloses both the actual indication of the location referred to, which may be tagged using the <ref> or <ptr> elements, and any accompanying material which gives more information about why the reader is being referred there.
Part: additional tag set for printed dictionaries
Member of classes: dictionaries, dictionaryParts, dictionaryTopLevel
DTD file: teidict2
Data description: May contain character data and phrase-level elements; usually contains a <ref> or a <ptr> element.
Occurs within: entry etym hom re sense trans
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang lbl link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title usg val view w xptr xref
Declaration:

```
<!ELEMENT xr            - O  (%paraContent | usg | lbl)*    >

<!ATTLIST xr                 %a.global
                             %a.dictionaries
          type               CDATA              #IMPLIED    >
```

Discussed in: 12.3.5.3 ('Cross References to Other Entries') on p. 293

xref

<xref> (i.e. extended reference) defines a reference to another location in the current document, or an external document, using an extended pointer notation, possibly modified by additional text or comment.

Attributes: [None: global and inherited attributes only.]
Part: additional tag set for analysis and interpretation
Member of classes: loc, terminologyInclusions, xPointer [and indirectly also:] phrase, pointer
DTD file: teilink2.dtd
Occurs within: abbr activity actor add addrLine addName admin affiliation author authority bibl biblScope birth bloc byline camera caption case castItem catDesc cell channel cit cl classCode closer colloc constitution corr country creation damage dataDesc date dateRange def del derivation desc descrip distance distinct distributor docAuthor docDate docEdition docImprint domain edition editor education emph equiv etym expan extent factuality figDesc firstLang foreign forename form fsDescr funder fw fDescr gen genName gloss gram gramGrp head headItem headLabel hi hyph imprimatur interaction item itype l label lang language langKnown lbl lem locale measure meeting mentioned mood name nameLink note num number occasion occupation opener orgdivn orgtitle orgtype orgName orig orth otherForm p per persName phr placeName pos preparedness principal pron publisher pubPlace purpose q quote rdg re ref reg region rendition residence resp restore role roleDesc roleName rs s salute seg sense set settlement sic signed socecStatus sound soCalled speaker sponsor stage street stress subc supplied surname syll symbol tagUsage tech term termEntry time timeRange title titlePart tns tr trailer trans u unclear usg valDesc view wit witness witDetail writing xr xref
May contain: #PCDATA abbr add address anchor att bibl biblFull biblStruct c caesura camera caption castList cit cl corr date dateRange dateStruct del distinct emph expan figure foreign formula gap gi gloss handShift hi label lang link list listBibl m measure mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote ref reg rs s seg sic sound soCalled stage table tag tech term text time timeRange timeStruct title val view w xptr xref
Declaration:

```
<!ELEMENT xref          - O  (%paraContent)                >
<!ATTLIST xref               %a.global
                             %a.xPointer                   >
```

Discussed in: 14.2 ('Extended Pointers') on p. 340; 13.2 ('Tags for Terminological Data') on p. 312

year

<year> (i.e. year) the year component of a date.

Attributes: [None: global and inherited attributes only.]
Example:

```
<date value='14-05-1993'>
<day type=name>Friday</day>,
<day type=number>14</day>
<month type=name>May</month>
<year>1993</year></date>
```

Part: additional tag set for
Member of classes: temporalExpr
DTD file: teind2
Occurs within: dateStruct timeStruct
May contain: #PCDATA
Declaration:

```
<!ELEMENT year          - -  (#PCDATA)                     >
<!ATTLIST year               %a.global
                             %a.temporalExpr               >
```

Discussed in: 20.4 ('Dates and Time') on p. 499

Part VIII

Reference Material

Chapter 36

Obtaining the TEI DTD

Full copies of the TEI DTD fragments are not repeated in this document, although they may be reconstituted by a careful reading of each chapter. The DTD fragments are widely available over the Internet and elsewhere. At the time of publication, the following anonymous file servers are expected to carry copies of a full set. The first is a Listserv server, the others are anonymous FTP servers.

- listserv@uicvm.uic.edu in filelist tei-l
- sgml1.ex.ac.uk in directory tei/p2/dtds
- ftp.ifi.uio.no in directory TEI

The files are these:

- *teiana2.dtd*: tags for simple analysis
- *teiana2.ent*: parameter entities for simple analysis
- *teiback2.dtd*: tags for back matter
- *teicert2.dtd*: tags for certainty and responsibility
- *teiclas2.ent*: parameter entities for the core tag sets
- *teicore2.dtd*: core tags
- *teicorp2.dtd*: tags for additional header material for corpora and collections
- *teidict2.dtd*: tags for printed dictionaries
- *teidict2.ent*: parameter entities for printed dictionaries
- *teidram2.dtd*: tags for performance texts
- *teidram2.ent*: parameter entities for performance texts
- *teifig2.dtd*: tags for tables, formulae, and graphics
- *teifig2.ent*: parameter entities for tables, formulae, and graphics
- *teifron2.dtd*: tags for front matter
- *teifsd2.dtd*: feature system declaration
- *teifs2.dtd*: tags for feature structure annotation
- *teigen2.dtd*: tags for general base
- *teigis2.ent*: parameter entities for names of TEI elements
- *teihdr2.dtd*: tags for TEI header
- *teikey2.ent*: parameter entities for TEI keywords
- *teilink2.dtd*: tags for linking, segmentation, and alignment
- *teilink2.ent*: parameter entities for linking, segmentation, and alignment
- *teimix2.dtd*: tags for mixed base
- *teind2.dtd*: tags for names and dates
- *teind2.ent*: parameter entities for names and dates
- *teinet2.dtd*: tags for networks, graphs, digraphs, and trees
- *teipl2.dtd*: tags for concurrent markup of pages and lines
- *teipros2.dtd*: tags for prose
- *teishd2.dtd*: tags for independent header
- *teispok2.dtd*: tags for spoken texts
- *teispok2.ent*: parameter entities for spoken texts

- *teistr2.dtd*: tags for default text structure
- *teitc2.dtd*: tags for text criticism
- *teitc2.ent*: parameter entities for text criticism
- *teiterm2.dtd*: tags for terminological data
- *teiterm2.ent*: parameter entities for terminological data
- *teite2f.dtd*: tags for terminological data
- *teite2n.dtd*: tags for terminological data
- *teitran2.dtd*: tags for transcription of primary sources
- *teitran2.ent*: parameter entities for transcription of primary sources
- *teitsd2.dtd*: tags for tag set documentation
- *teivers2.dtd*: tags for verse
- *teivers2.ent*: parameter entities for verse
- *teiwsd2.dtd*: tags for writing system declaration
- *tei2.dtd*: main TEI DTD file

Chapter 37

Obtaining TEI WSDs

The TEI has produced a number of exemplary Writing System Definitions, corresponding with several major international standards. This section lists those which are currently available. In each case, the root name specified is that used for the file on the TEI file servers.

A typical public identifier for a Writing System Declaration from this list would be

```
<!ENTITY xxxx PUBLIC
    " -//TEI-1994//NOTATION WSD for YYY //EN" >
```

where xxxxx corresponds with the short name given in the list below, and YYY with the longer description provided.

Writing system declarations corresponding to some standard character sets include:

ISO646ss ISO 646 Non-national Subset
ISO646ir ISO 646 International Reference Version
ISO88591 ISO 8859 Part 1 (Latin 1, Western Europe)
ISO88592 ISO 8859 Part 2 (Latin 2, Eastern Europe)
ISO88595 ISO 8859 Part 5 (Latin and Cyrillic)
ISO88597 ISO 8859 Part 7 (Latin and Greek)
ISO88598 ISO 8859 Part 8 (Latin and Hebrew)
ISO88599 ISO 8859 Part 9 (Latin 5, Western Europe and Turkey)

Writing system declarations documenting special sets of entities for different alphabets, or commonly used transliteration systems, include the following:

TEIarb
- TEI Entities for Arabic

TEIcop
- TEI Entities for Coptic

TEIgrk
- TEI Entities for Greek

TEIipa
- TEI Entities for International Phonetic Alphabet

All standard TEI writing system declarations may be obtained from the same file servers as are listed in chapter 36 (‘Obtaining the TEI DTD’) on p. 983.

Chapter 38

Sample Tag Set Documentation

38.1 Tag Documentation for the TEI P Element

This example shows the documentation for the element <p>. Since this element has no attributes other than global attributes, its <attList> element is empty. For a formatted, printed version of this information, see the alphabetical reference section.

```
<tagDoc id=p usage=req>
<gi>p</gi>
<rs>paragraph</rs>
<desc>marks paragraphs in prose.</desc>
<attlist>
</attlist>
<exemplum><eg><![ CDATA [
<p>Hallgerd was outside.  <q>There is blood on your
axe,</q> she said.  <q>What have you done?</q></p>
<p><q>I have now arranged that you can be married a
second time,</q> replied Thjostolf.</p>
<p><q>Then you must mean that Thorvald is dead,</q>
she said.</p>
<p><q>Yes,</q> said Thjostolf.  <q>And now you must
think up some plan for me.</q>
]]>
</eg>
</exemplum>
<remarks><p>In some contexts, the paragraph may have a specialized
meaning, e.g. in the tag set for dictionaries, <gi>p</gi> is used to
enclose any running text, and thus does not imply text set off as is
conventionally done in running prose.
<part type='top' name='core'>additional tag set for common core
features</part>
<classes names='chunk'>chunk [and indirectly also:] common</classes>
<files names='teicore2'>
<datadesc>May contain character data and phrase-level elements.
<parents>add argument availability body broadcast castList corr
correction div div0 div1 div2 div3 div4 div5 div6 div7 editionStmt
editorialDecl encodingDesc epigraph epilogue equipment equiv exemplum
figure hyphenation interpretation item langUsage metDecl
normalization note particDesc particLinks performance person
personGrp projectDesc prologue publicationStmt q quotation quote
recording recordingStmt refsDecl remarks samplingDecl scriptStmt
segmentation seriesStmt set setting settingDesc sic sourceDesc sp
stage stdVals view</parents>
<children>#PCDATA abbr add address anchor att bibl biblFull
```

```
biblStruct c caesura camera caption castList cit cl corr date
dateRange dateStruct del distinct emph expan figure foreign formula
gap gi gloss handShift hi label lang link list listBibl m measure
mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote
ref reg rs s seg sic sound soCalled stage table tag tech term text
time timeRange timeStruct title val view w xptr xref</children>
<elemdecl>
<![ CDATA [
<!ELEMENT p             - O  (%paraContent;)                    >
]]>
</elemdecl>
<attldecl>
<![ CDATA [
<!ATTLIST p                  %a.global                          >
]]>
</attldecl>

<ptr type=div2 target=COpa>
<ptr type=div2 target=DRpal>
</tagdoc>
```

38.2 Tag Documentation for the TEI HI Element

The <hi> element, shown below, offers an example of attribute definition. Since the attribute definition in question overrides a definition inherited from the *global* class, no reference is made to the entity *a.global*. Instead, the other global attributes are all redefined.

```
<tagdoc usage=opt id="hi"><gi>hi</gi>
<rs>highlighted</rs>
<desc>marks a word or phrase as graphically distinct from the
surrounding text, for reasons concerning which no claim is made. </desc>
<attlist>
<!-- ................................................. rend -->
<attDef usage=rec>
<attname>rend
<rs>rendition</rs>
<desc>describes the rendition or presentation of the word or phrase
highlighted.
<datatype>CDATA
<valdesc>a set of keywords from some suitable vocabulary; no systematic
recommendations for such a vocabulary are made by these Guidelines.
<default>#IMPLIED
<eg><![ CDATA [

]]>
</eg>
<remarks><p>The <att>rend</att> attribute, though optional in general,
is recommended on the <gi>hi</gi> element.
</attDef>
</attlist>
<exemplum><eg><![ CDATA [
<hi rend=gothic>And this Indenture further witnesseth</hi>
that the said <hi rend=italic>Walter Shandy</hi>, merchant,
in consideration of the said intended marriage ...
]]>
</eg></exemplum>
<remarks></remarks>
<part type='aux' name='hdr'>auxiliary tag set for TEI headers</part>
<classes names='hqphrase'>hqphrase [and indirectly also:]
```

```
phrase</classes>
<files names='teicore2'>
<datadesc>free prose</datadesc>
<parents>abbr activity actor add addrLine addName admin affiliation
author authority bibl biblScope birth bloc byline camera caption case
castItem catDesc cell channel cl classCode closer colloc constitution
corr country creation damage dataDesc date dateRange def del
derivation desc descrip distance distinct distributor docAuthor
docDate docEdition docImprint domain edition editor education emph
equiv etym expan extent factuality figDesc firstLang foreign forename
form fsDescr funder fw fDescr gen genName gloss gram gramGrp head
headItem headLabel hi hyph imprimatur interaction item itype l label
lang language langKnown lbl lem locale measure meeting mentioned mood
name nameLink note num number occasion occupation opener orgdivn
orgtitle orgtype orgName orig orth otherForm p per persName phr
placeName pos preparedness principal pron publisher pubPlace purpose
q quote rdg re ref reg region rendition residence resp restore role
roleDesc roleName rs s salute seg sense set settlement sic signed
socecStatus sound soCalled speaker sponsor stage street stress subc
supplied surname syll symbol tagUsage tech term time timeRange title
titlePart tns tr trailer trans u unclear usg valDesc view wit witness
witDetail writing xr xref</parents>
<children>#PCDATA abbr add address anchor att bibl biblFull
biblStruct c caesura camera caption castList cit cl corr date
dateRange dateStruct del distinct emph expan figure foreign formula
gap gi gloss handShift hi label lang link list listBibl m measure
mentioned move name note num orig oRef oVar phr ptr pRef pVar q quote
ref reg rs s seg sic sound soCalled stage table tag tech term text
time timeRange timeStruct title val view w xptr xref</children>
<elemdecl>
<![ CDATA [
<!ELEMENT hi            - -  (%paraContent;)                  >
]]>
</elemdecl>
<attldecl>
<![ CDATA [
<!ATTLIST hi                 %a.fs
                             %a.analysis
                             %a.linking
                             %a.terminology
          id                 ID                  #IMPLIED
          n                  CDATA               #IMPLIED
          lang               IDREF               %INHERITED

          rend               CDATA               #IMPLIED      >
]]>
</attldecl>

<ptr target=COhqhe>
<ptr target=COhqh>
</tagdoc>
```

38.3 Tag Documentation for the TEI Div Element

The example below shows the documentation for the element; this element is a member of the class *divn*, the documentation for which is shown further below.

```
<tagdoc usage=rwa id="div"><gi>div</gi>
<rs>text division</rs>
```

```
<desc>contains a subdivision of the front, body, or back of a
text.</desc>
<attlist>
</attlist>
<exemplum><eg><![ CDATA [
<body>
<div type=part><head>Fallacies of Authority</head>
<epigraph>The subject of which is Authority in
various shapes, and the object, to repress all
exercise of the reasoning faculty.</epigraph>
  <div type=chapter n=1><head>The Nature of Authority</head>
  <p>With reference to any proposed measures
  having for their object the greatest happiness
  of the greatest number....
     <div type=section n='1.1'><head>Analysis of Authority
      <p>What on any given occasion is the legitimate
      weight or influence to be attached to authority
      <!-- ... -->
     </div>
     <div type=section n='1.2'><head>Appeal to Authority,
        in What Cases Fallacious.
      <p>Reference to authority is open to the charge of
      fallacy when...
      <!-- ... -->
     </div>
     <!-- other sections here -->
   </div>
   <div type=chapter n=2>
   <!-- other chapters here -->
   </div>
</div>
<!-- other parts here -->
</body>
]]><!-- J Bentham: Handbook of Political Fallacies (1824), ed -->
<!-- Larrabee, Johns Hopkins Pr, 1952 -->
</eg></exemplum>
<remarks>
<part type='base' name='core'>base tag set for common core
features</part>
<classes names='divn decling'>declaring, divn</classes>
<files names='teistr2'>
<datadesc>any sequence of low-level structural elements, possibly grouped
into lower subdivisions.</datadesc>
<parents>back body div front</parents>
<children>argument bibl biblFull biblStruct byline camera caption
castList cit closer div docAuthor docDate entry entryFree epigraph
event eTree graph head kinesic l label lg lg1 list listBibl move note
opener p pause q quote salute shift signed sound sp stage superentry
tech termEntry trailer tree u view vocal writing</children>
<elemdecl>
<![ CDATA [
<!ELEMENT div           - O  ((%m.divtop;)*, (div+ |
                             ((%component;)+, div*)),
                             (%m.divbot;)*)                    >
]]>
</elemdecl>
<attldecl>
<![ CDATA [
<!ATTLIST div                %a.global
                             %a.declaring
```

```
                                %a.divn                                 >
]]>
</attldecl>

<ptr target=DSdiv>
<ptr target=tedt><ptr target=dsdiv1>
</tagdoc>
```

38.4 Class Documentation for the TEI Divn Class

The example below shows the documentation for the class *divn*, which is referred to by the documentation for shown above. The class documentation in the alphabetical reference section of this document uses a slightly extended tag set which provides for information about elements within which members of the class may appear, members of the class, and the actual declaration of the parameter entities by which properties of the class are expressed.

```
<classDoc id=divn type=atts>
<class>divn</class>
<desc>structural elements which behave in the same way as divisions.
<attlist>
<attDef usage=rec>
<attname>type</attname>
<desc>specifies a name conventionally used
for this level of subdivision, e.g. <q>act</q>,
<q>volume</q>, <q>book</q>, <q>section</q>, <q>canto</q>, etc.</desc>
<datatype>CDATA
<valdesc>any string of characters</valdesc>
<default>#CURRENT
</attdef>
<attdef usage=opt><attname>org
<desc>specifies how the content of the division is organized.
<datatype>(composite | uniform)
<vallist type=closed>
<val>composite<desc>composite content: i.e. no claim is
made about the sequence in which the immediate contents of this
division are to be processed, or their inter-relationships.</desc>
<val>uniform<desc>uniform content: i.e. the immediate contents
of this element are regarded as forming a logical unit,
to be processed in sequence.</desc>
</vallist>
<default>uniform
</attdef>
<attdef>
<attname>sample
<desc>indicates whether this division is a sample of the original source
and if so, from which part.
</desc>
<datatype>(initial | medial | final | unknown | complete)
<vallist type=closed>
<val>initial<desc>division lacks material present at end in
source.</desc>
<val>medial<desc>division lacks material at start and end.</desc>
<val>final<desc>division lacks material at start.</desc>
<val>unknown<desc>position of sampled material within original
unknown.</desc>
<val>complete<desc>division is not a sample.</desc>
</vallist>
<default>complete
<remarks>
```

```
</attDef>
<attdef usage=mwa><attname>part</attname>
<desc>specifies whether or not the division is fragmented by some other
structural element, for example a speech which is divided between two
or more verse stanzas.
</desc>
<datatype>(Y | N | I | M | F)
<vallist type=closed>
<val>Y<desc>the division is incomplete in some respect</desc>
<val>N<desc>either the division is complete, or no claim is made as to
its completeness.</desc>
<val>I<desc>the initial part of an incomplete division</desc>
<val>M<desc>a medial part of an incomplete division</desc>
<val>F<desc>the final part of an incomplete division</desc>
</vallist>
<default>N
<remarks><p>The values I, M, or F should be used only where it is
clear how the division is to be reconstituted.
</attdef>
</attlist>
<part type='base' name='core'>base tag set for common core
features</part>
<classes names=''></classes>
<files names='teiclas2.ent'>
<ptr target=DS>
</classdoc>
```

Chapter 39

Formal Grammar for the TEI-Interchange-Format Subset of SGML

This grammar is intended to help make SGML more comprehensible for formal manipulation. For this reason, a number of simplifications have been undertaken, which are described below in section 39.8 ('Differences from ISO 8879') on p. 1004. These simplifications may cause this grammar to accept some documents not accepted by the official grammar. As far as is known, however, the grammar provided here will recognize any valid SGML document in the TEI Interchange Format.

For ease in relating this grammar to the formal grammar defined in ISO 8879, comments for each group here give the numbers of the related productions in that grammar. Where the changes to SGML syntax suggested by the SGML working group in its document ISO/IEC JTC1 / SC18 / WG8 / N1035 would affect the productions here, that fact is noted after the affected production.

Each sub-grammar given here has been checked for ambiguity with *bison*, a public-domain workalike for *yacc*, and *flex*, a public-domain workalike for *lex*. The bison and flex source files, including the simple modifications needed to implement the recognition-mode stack, are available from the TEI.

The SGML declaration grammar and the DTD grammar have no ambiguities. The document grammar has several ambiguities, which are discussed in section 39.5 ('Grammar for Document Instance') on p. 1000.

39.1 Notation

The notation used here is based on notations commonly used in writing context-free grammars. All non-terminals are written as single tokens.

`::=`
- "is defined as"

`a b`
- instance of `a` followed by instance of `b`, possibly separated by white space; white space may be required if `a` and `b` cannot otherwise be delimited

`a | b`
- instance of `a` or instance of `b`

`a || b`
- instance of `a` followed immediately by an instance of `b`, without any intervening white space

`/* */`
- comment; right-hand sides comprising only a comment are used to call attention to produc-

tions in which the left-hand sides can reduce to the empty string

//

- comment; runs to end of line

All non-quoted strings are non-terminals. All terminals are quoted.

39.2 Grammar for SGML Document (Overview)

An SGML document is preceded by an SGML declaration and a prolog comprising one or more document-type declarations. It may be accompanied by one or more subdocument entities, text entities, non-SGML entities, etc., but for simplicity these last are not discussed here.

```
SGMLdoc ::= SGMLdeclaration prolog docinstance  // cf. 1, 2
```

The grammars for the SGML declaration, the prolog, and the document instance are provided in the following three sections.

39.3 Grammar for SGML Declaration

This grammar is substantively the same as that in ISO 8879; it does not reflect the restrictions placed on SGML declarations for TEI-conformant documents.

```
SGMLdeclaration ::= '<!SGML'                // cf. 171
                    'ISO 8879:1986'
                    'CHARSET' charset       // cf. 172
                    'CAPACITY' capacity     // cf. 180
                    'SCOPE' scope           // cf. 181
                    'SYNTAX' syntax
                    'FEATURES'              // cf. 195-198
                        'MINIMIZE'
                                'DATATAG'  yesno
                                'OMITTAG'  yesno
                                'RANK'     yesno
                                'SHORTTAG' yesno
                        'LINK'
                                'SIMPLE' count
                                'IMPLICIT' yesno
                                'EXPLICIT' count
                        'OTHER'
                                'CONCUR' count
                                'SUBDOC' count
                                'FORMAL' yesno
                    'APPINFO' appinfo       // cf. 199
                    '>'
/*
** Strictly speaking, the blank in 'ISO 8879:1986' may be
** replaced by any amount or type of white space.
*/

charset     ::= baseset descset             // cf. 173
              | charset baseset descset
baseset     ::= 'BASESET' pubid             // cf. 174
/* For pubid, see Common Constructs below */
descset     ::= 'DESCSET' chardesc          // cf. 175
              |  descset  chardesc
chardesc    ::= NUMBER NUMBER NUMBER        // cf. 176-179
              | NUMBER NUMBER LITERAL
              | NUMBER NUMBER 'UNUSED'
```

```
/*
** The first two numbers in chardesc identify starting point
** and a number of characters in the described character set;
** the third number gives the starting position for that run
** in the base set; a literal provides a description of the
** character(s); 'UNUSED' marks the run as unused.
*/

capacity    ::= 'PUBLIC' pubid              // cf. 180
              | 'SGMLREF' caplist
caplist     ::= capname NUMBER
              | caplist capname NUMBER
capname     ::= 'TOTALCAP' | 'ENTCAP'   | 'ENTCHCAP'
              | 'ELEMCAP'  | 'GRPCAP'   | 'EXGRPCAP'
              | 'EXNMCAP'  | 'ATTCAP'   | 'ATTCHCAP'
              | 'AVGRPCAP' | 'NOTCAP'   | 'NOTCHCAP'
              | 'IDCAP'    | 'IDREFCAP' | 'MAPCAP'
              | 'LKSETCAP' | 'LKNMCAP'      // cf. Fig. 5

scope       ::= 'DOCUMENT' | 'INSTANCE'     // cf. 181

syntax      ::= 'PUBLIC' pubid              // cf. 182-183
              | 'PUBLIC' pubid switchlist
              | 'SHUNCHAR' shunchars        // cf. 184
                charset                     // cf. 185
                'FUNCTION'                  // cf. 186
                    'RE' NUMBER 'RS' NUMBER 'SPACE' NUMBER
                    funlist
                'NAMING'                    // cf. 189
                    'LCNMSTRT' LITERAL
                    'UCNMSTRT' LITERAL
                    'LCNMCHAR' LITERAL
                    'UCNMCHAR' LITERAL
                    'NAMECASE' 'GENERAL' yesno 'ENTITY' yesno
                'DELIM'                     // cf. 190-92
                    'GENERAL' 'SGMLREF' gendelim
                    'SHORTREF' srdelim
                'NAMES'    'SGMLREF' nameset // cf. 193
                'QUANTITY' 'SGMLREF' quantityset // cf. 194
/*
** Document WG8 / N1035 suggests allowing multiple literals
** after each keyword in the NAMING section; this may be
** effected by adding the non-terminal literalset after each
** LITERAL.
*/

switchlist  ::= 'SWITCHES' NUMBER NUMBER    // cf. 183
              | switchlist NUMBER NUMBER

shunchars   ::= 'NONE'                      // cf. 184
              | shunlist
shunlist    ::= 'CONTROLS'
              | NUMBER
              | shunlist NUMBER

funlist     ::= NAME funclass NUMBER        // cf. 186-187
              | funlist NAME funclass NUMBER
funclass    ::= 'FUNCHAR' | 'MSICHAR' | 'MSOCHAR' // cf. 188
              | 'MSSCHAR' | 'SEPCHAR'
```

```
gendelim    ::= /* nil */                    // cf. 191
              | gendelim delimname LITERAL
delimname   ::= 'AND'   | 'COM'  | 'CRO'    // cf. Fig. 3
              | 'DSC'   | 'DSO'  | 'DTGC'   // clause 9.6
              | 'DTGO'  | 'ERO'  | 'ETAGO'
              | 'GRPC'  | 'GRPO' | 'LIT'
              | 'LITA'  | 'MDC'  | 'MDO'
              | 'MINUS' | 'MSC'  | 'NET'
              | 'OPT'   | 'OR'   | 'PERO'
              | 'PIC'   | 'PIO'  | 'PLUS'
              | 'REFC'  | 'REP'  | 'RNI'
              | 'SEQ'   | 'SHORTREF'
              | 'STAGO' | 'TAGC' | 'VI'

srdelim     ::= 'SGMLREF' literalset        // cf. 192
              | 'NONE' literalset
literalset  ::= /* nil */
              | literalset LITERAL
nameset     ::= /* nil */
              | nameset sgmlname NAME
/*
** WG8 / N1035 substitutes a LITERAL for the NAME of the
** preceding rule; the value must be a NAME in the declared
** concrete syntax, but it need not be a legal name in the
** reference concrete syntax.
*/

sgmlname    ::= 'ANY'      | 'ATTLIST'  | 'CDATA'
              | 'CONREF'   | 'CURRENT'  | 'DEFAULT'
              | 'DOCTYPE'  | 'ELEMENT'  | 'EMPTY'
              | 'ENDTAG'   | 'ENTITIES' | 'ENTITY'
              | 'FIXED'    | 'ID'       | 'IDLINK'
              | 'IDREF'    | 'IDREFS'   | 'IGNORE'
              | 'IMPLIED'  | 'INCLUDE'  | 'INITIAL'
              | 'LINK'     | 'LINKTYPE' | 'MD'
              | 'MS'       | 'NAME'     | 'NAMES'
              | 'NDATA'    | 'NMTOKEN'  | 'NMTOKENS'
              | 'NOTATION' | 'NUMBER'   | 'NUMBERS'
              | 'NUTOKEN'  | 'NUTOKENS' | 'O'
              | 'PCDATA'   | 'PI'       | 'POSTLINK'
              | 'PUBLIC'   | 'RCDATA'   | 'RE'
              | 'REQUIRED' | 'RESTORE'  | 'RS'
              | 'SDATA'    | 'SHORTREF' | 'SIMPLE'
              | 'SPACE'    | 'STARTTAG' | 'SUBDOC'
              | 'SYSTEM'   | 'TEMP'     | 'USELINK'
              | 'USEMAP'

quantityset ::= /* nil */
              | quantityset quantity NUMBER
quantity    ::= 'ATTCNT'   | 'ATTSPLEN' | 'BSEQLEN' // Cf.
              | 'DTAGLEN'  | 'DTEMPLEN' | 'ENTLVL'  // Fig. 6
              | 'GRPCNT'   | 'GRPGTCNT' | 'GRPLVL'
              | 'LITLEN'   | 'NAMELEN'  | 'NORMSEP'
              | 'PILEN'    | 'TAGLEN'   | 'TAGLVL'

yesno       ::= 'NO' | 'YES'
count       ::= 'NO' | 'YES' NUMBER

appinfo     ::= 'NONE' | LITERAL            // cf. 199
/*
```

```
** The literal string is restricted to letters, digits,
** whitespace, and 'specials':  viz. any of
** ' ( ) + , - . / : = ?
*/
```

39.4 Grammar for DTD

An SGML *prolog* is composed of one or more document type declarations; if multiple DTDs are present, the SGML declaration must include "CONCUR YES" in the FEATURES section. A document type declaration names the root element of the document and declares (in an external file, in a DTD subset, or both) elements, attributes, notations, and entities used in the document instance. Interspersed with these declarations may be comments and processing instructions.

```
prolog      ::= misc dtdseq                 // cf. 7, 9

/* For misc, see Common Constructs below */

dtdseq      ::= dtd misc                    // cf. 7
              | dtdseq dtd misc

                                            // cf. 110, 111, 30
dtd         ::= '<!DOCTYPE' NAME extid '[' dtdsubset ']' '>'
              | '<!DOCTYPE' NAME       '[' dtdsubset ']' '>'
              | '<!DOCTYPE' NAME extid                  '>'
              | '<!DOCTYPE' NAME                        '>'

dtdsubset   ::= /* */                       // cf. 112, 113, 114
              | dtdsubset elementdecl
              | dtdsubset attlistdecl
              | dtdsubset notationdecl
              | dtdsubset entitydecl
              | dtdsubset commdecl
              | dtdsubset procinst

/* Element Declarations */                  // cf. 116
elementdecl ::= '<!ELEMENT' elemtype minimiz contentdecl '>'

elemtype    ::= NAME                        // cf. 117, 30, 72
              | '(' namegrp ')'

namegrp     ::= andnames                    // cf. 69, 131
              | ornames
              | seqnames
seqnames    ::= NAME
              | seqnames ',' NAME
ornames     ::= NAME     '|' NAME
              | ornames  '|' NAME
andnames    ::= NAME     '&' NAME
              | andnames '&' NAME

minimiz     ::= min min                     // cf. 122-124
min         ::= 'O' | '-'

contentdecl ::= 'CDATA'                     // cf. 125, 126
              | 'RCDATA'
              | 'EMPTY'
              | 'ANY'  exceptions
              | model  exceptions
```

```
model       ::= '(' tokengrp ')'            // cf. 127
              | '(' tokengrp ')?'
              | '(' tokengrp ')*'
              | '(' tokengrp ')+'

tokengrp    ::= seqtokens                   // cf. 127
              | ortokens
              | andtokens
seqtokens   ::= token
              | seqtokens ',' token
ortokens    ::= token     '|' token
              | ortokens  '|' token
andtokens   ::= token     '&' token
              | andtokens '&' token

token       ::= '#PCDATA'                   // cf. 128-130
              | NAME || occurrence
              | model

occurrence  ::= /* nil */                   // cf. 132
              | '?'
              | '*'
              | '+'

exceptions  ::= exclusions inclusions       // cf. 138-140
exclusions  ::= /* nil */
              | '-(' namegrp ')
inclusions  ::= /* nil */
              | '+(' namegrp ')'


/* Attribute List Declarations */
attlistdecl ::= '<!ATTLIST' associated attdeflist '>' // Cf. 141

associated  ::= elemtype
              | assocnotatn

assocnotatn ::= '#NOTATION' NAME            // Cf. 149.1
              | '#NOTATION' '(' namegrp ')'

attdeflist  ::= attdef                      // Cf. 142
              | attdeflist attdef

attdef      ::= NAME valtype default        // Cf. 143-44

valtype     ::= 'CDATA'   |                 // Cf. 145
              | 'ENTITY'  | 'ENTITIES'
              | 'ID'      |
              | 'IDREF'   | 'IDREFS'
              | 'NAME'    | 'NAMES'
              | 'NMTOKEN' | 'NMTOKENS'
              | 'NUMBER'  | 'NUMBERS'
              | 'NUTOKEN' | 'NUTOKENS'
              | 'NOTATION' '(' namegrp ')'
              | '(' nmtokgrp ')'

nmtokgrp    ::= nmtokcom | nmtokbar | nmtokamp   // Cf. 68, 131
nmtokcom    ::= nametoken
              | nmtokcom  ',' nametoken
nmtokbar    ::= nametoken '|' nametoken
```

```
              | nmtokbar  '|' nametoken
nmtokamp    ::= nametoken '&' nametoken
              | nmtokamp  '&' nametoken
nametoken   ::= NAME
              | NUMBER
              | NUMTOKEN

default     ::=  value                      // Cf. 147
              | '#FIXED' value
              | '#REQUIRED'
              | '#CURRENT'
              | '#CONREF'
              | '#IMPLIED'

/* For value, see Common Constructs below */


/* Notation Declarations */                 // cf. 148-49, 41
notationdecl ::= '<!NOTATION' NAME extid '>'


/* Entity Declarations */                   // cf. 101-04
entitydecl  ::= '<!ENTITY' NAME       enttext '>'
              | '<!ENTITY' '#DEFAULT' enttext '>'
              | '<!ENTITY' '% ' NAME  enttext '>'
/*
** Strictly, any white space is acceptable after the %
** in a parameter entity declaration, not just a single
** space.
*/

enttext     ::= LITERAL                     // cf. 105-08
              | 'CDATA'    LITERAL
              | 'SDATA'    LITERAL
              | 'PI'       LITERAL
              | 'STARTTAG' LITERAL
              | 'ENDTAG'   LITERAL
              | 'MS'       LITERAL
              | 'MD'       LITERAL
              | extid enttype

enttype     ::= /* */                       // cf. 108-109, 149.2
              | 'SUBDOC'
              | 'CDATA' NAME
              | 'CDATA' NAME '[' attspecset ']'
              | 'NDATA' NAME
              | 'NDATA' NAME '[' attspecset ']'
              | 'SDATA' NAME
              | 'SDATA' NAME '[' attspecset ']'
/* For attspecset, see Common Constructs below. */

extid       ::= 'SYSTEM'                    // cf. 73
              | 'SYSTEM' sysid
              | 'PUBLIC' pubid
              | 'PUBLIC' pubid sysid
/* For pubid, see Common Constructs below */

sysid       ::= LITERAL                     // cf. 75
```

39.5 Grammar for Document Instance

The SGML document instance is composed of one element (the *root element*), followed optionally by white space, comments, and processing instructions. The root element, like any other, has a start-tag, content, and an end-tag, or only a start-tag (if it is empty). The specific sort of content recognized within an element depends upon its element declaration.

```
docinstance ::= element misc                // cf. 10-12

element     ::= start-tag content end-tag   // cf. 13
              | start-tag
/*
** N.B. this BNF is for minimal SGML documents; tags may
** not be omitted.
*/

start-tag   ::= '<' || NAME attspecset '>'  // cf. 14, 28-30
              | '<(' namegrp ')' || NAME attspecset '>'
/*
** The name group is used only if the SGML declaration
** specifies CONCUR YES.
** For attspecset, see Common Constructs below.
*/

content     ::= mixedcontent                // cf. 24
              | elemcontent
              | rcdata
              | cdata

mixedcontent ::= /* nil */                  // cf. 25
              | mixedcontent STRING
              | mixedcontent element
              | mixedcontent misccontent

elemcontent ::= /* nil */                   // cf. 26
              | elemcontent element
              | elemcontent misccontent
cdata       ::= STRING                      // cf. 47
              | cdata STRING
rcdata      ::= STRING                      // cf. 46
              | rcdata STRING
/*
** White space is ignored between elements in element content,
** but not in mixed content.  In CDATA and RCDATA, start-tag
** delimiters are not recognized.  In CDATA, entity reference
** delimiters are not recognized.  An element's content model
** determines whether it is scanned for mixed content, element
** content, CDATA content, or RCDATA content.
*/

misccontent ::= commdecl                    // cf. 27
              | procinst
/*
** For commdecl and procinst, see Common Constructs below.
** Omitted here for simplicity are short-reference and
** link-set use declarations and short references (which are
** not allowed in TEI interchange format), entity references
** (which are assumed to be handled by the lexical scanner),
** and marked-section declarations (also in lexical
** scanner).
*/
```

```
end-tag     ::= '</' || NAME '>'            // cf. 19, 21
              | '</(' namegrp ')' || NAME '>'
              | '</>'
/*
** The name group is used only if the SGML declaration
** specifies CONCUR YES.
** N.B. The last form (short end-tag) is not allowed in the
** TEI Interchange Format.
*/
```

The document-instance grammar just given contains two sets of formal ambiguities. One set concerns the distinction among mixed content, element content, RCDATA, and CDATA, which depends not on the document content but on the definition of the element within which they appear. These conflicts can be eliminated by eliminating the distinction and assigning the task of distinguishing content type (and alerting the lexical analyzer to modify its behavior) to the semantic rules of the parser, rather than to the syntax.

The second set of ambiguities arises in connection with start-tags: after a start-tag, empty elements are complete, others not, and the ambiguity can be resolved only by consulting the DTD, not by lookahead. Such conflicts can be avoided by defining document content as an unstructured sequence of start-tags, end-tags, and data content; the parser's semantic actions must enforce the pairing and nesting of start- and end-tags and the distinction between empty and non-empty elements. Despite the ambiguities, the grammar given here seems to express the nature of SGML documents more clearly than the unambiguous alternative and so has not been changed; the changes needed to eliminate the parsing conflicts are these:

- delete *element*, *mixedcontent*, *elemcontent*, *cdata*, and *rcdata*.
- redefine *docinstance* and *content* as follows:

```
docinstance ::= start-tag content           // cf. 10-12
content     ::= /* nil */                   // cf. 25
              | content STRING
              | content start-tag
              | content end-tag
              | content misccontent
```

Applications using this simplification must distinguish mixed content, element content, RCDATA, and CDATA using other methods than document syntax. They can check the appropriate matching of start- and end-tags using a simple element stack with provision for empty elements. Some SGML normalizers provide explicit end-tags for empty elements to simplify this task.

39.6 Common Syntactic Constructs

This section defines syntactic constructions used in more than one of the three preceding grammar fragments.

```
misc        ::= /* nothing */               // cf. 8
              | misc commdecl
              | misc procinst

commdecl    ::= '<!--' STRING '--' commseq '>'   // cf. 91, 92
              | '<!>'
commseq     ::= /* nil */
              | commseq '--' STRING '--'

procinst    ::= '<?' STRING '>'             // cf. 44

attspecset  ::= /* nil */                   // cf. 31
```

```
              | attspecset attspec

attspec     ::= NAME '=' value              // cf. 32
              | value
/*
** NAME may be omitted only if the attribute has an
** enumerated range of values and the value is an unquoted
** name token.
*/

value       ::= LITERAL                     // Cf. 33
              | NAME
              | NUMBER
              | NUMTOKEN

pubid       ::= LITERAL                     // cf. 74, 76
/*
** The literal string is restricted to letters, digits,
** whitespace, and 'specials':  viz. any of
** ' ( ) + , - . / : = ?
*/
```

39.7 Lexical Scanner

The grammar given above assumes a lexical scanner which

- scans for the terminal strings represented here in quotes
- scans for certain other token types (listed below)
- handles white space and some comments without returning them
- recognizes and expands entity references appropriately without notifying the parser

N.B. the literals given here for delimiters, keywords, and in the definitions of character classes and character types, are those used in the reference concrete syntax of SGML; a full SGML parser must be able to use other concrete syntaxes.

The token types to be returned by the lexical scanner include (in addition to the literals used in the grammars above):

- name
- number
- numtoken
- literal
- string

These are printed in all caps in the grammar and are defined thus:

```
NAME        ::= letter                      // Cf. 55
              | NAME || letter
              | NAME || digit
              | NAME || othernamech
NUMBER      ::= digit                       // cf. 56
              | NUMBER || digit
NUMTOKEN    ::= digit || letter             // cf. 58
              | digit || othernamech
              | NUMTOKEN || letter
              | NUMTOKEN || digit
              | NUMTOKEN || othernamech
LITERAL     ::= "'" || STRING || "'"        // Cf. 66, 76, 34
              | '"' || STRING || '"'
STRING      ::= /* */
              | STRING || character
```

```
character   ::= letter | digit | othercharacter
letter      ::= 'a' | 'b' | 'c' | 'd' ... | 'z'
              | 'A' | 'B' | 'C' | 'D' ... | 'Z'
digit       ::= '0' | '1' | '2' | '3' ... | '9'
othernamech ::= '-' | '.'
whitespace  ::= space | tab | record-end | record-start
space          ::= /* as defined in SGML declaration */
tab            ::= /* as defined in SGML declaration */
record-end     ::= /* as defined in SGML declaration */
record-start   ::= /* as defined in SGML declaration */
othercharacter ::= /* as defined in SGML declaration */
```

This list of primitive token types differs slightly from that of ISO 8879, which defines names, numbers, name tokens, and number tokens as overlapping sets of tokens distinguished by context. The redefinition provided here assigns each string to a single class and thus allows a simpler lexical analyzer. The terms in ISO 8879 correspond to those here in the following way:

- ISO 8879 *name* = NAME
- ISO 8879 *number* = NUMBER
- ISO 8879 *name token* = NAME | NUMBER | NUMTOKEN
- ISO 8879 *number token* = NUMBER | NUMTOKEN

The grammar given above assumes that the lexical scanner will recognize and handle entity references and marked sections. Entity references take one of the following forms (parameter entities within the DTD and within marked section declarations, general entities within document content and attribute values):

```
entityref ::= '&' NAME erc
            | '&' '(' namegrp ')' erc
peref     ::= '%' NAME erc
            | '%' '(' namegrp ')' erc
/*
** The name groups are used only if the SGML declaration
** specifies CONCUR YES.  The TEI Interchange Format
** specifies that entities should never be defined
** differently in different DTDs, so the name group form
** is strictly speaking unnecessary for that format.
*/
erc       ::= ';'
            | '&#RE;'                       // i.e. record-end
            | /* nothing */
/*
** If the entity reference is not ended explicitly by a
** semicolon or end of line, it is ended implicitly by the
** first non-name character.  Unlike the semicolon or
** record-end, this non-name character counts as data and is
** passed to the application as data.
*/
```

The processing of an entity reference may involve scanning its replacement text for delimiters, passing its content to the parser without scanning for delimiters, opening a new file if the entity is external to the SGML document, and other special processing not described here.

Marked sections take the following forms (n.b. the marked section keywords may be replaced by parameter entity references).

```
/* Marked Sections */                       // Cf. 97-100
msignore  ::= '<![' ignore   '[' anything  ']]' || '>'
mscdata   ::= '<![' kwcdata  '[' chardata  ']]' || '>'
msrcdata  ::= '<![' kwrcdata '[' rchardata ']]' || '>'
msinclude ::= '<![' include  '[' scandata  ']]' || '>'

/* Marked Section keywords */
ignore    ::= kwcdata 'IGNORE' kwcdata
```

```
            | ignore  'IGNORE' kwcdata
kwcdata   ::= kwrcdata 'CDATA' kwrcdata
            | kwcdata  'CDATA' kwrcdata
kwrcdata  ::= include  'CDATA' include
            | kwrcdata 'CDATA' include
include   ::= /* nothing */
            | include 'INCLUDE'
            | include 'TEMP'
/*
** Multiple keywords may appear; rank order is IGNORE,
** CDATA, RCDATA, INCLUDE.  TEMP is also legal but has
** no effect.
*/

chardata  ::= /* characters returned to parser as
                 character data, regardless of what
                 delimiters are present */
rchardata ::= /* characters scanned for entity references,
                 and then returned to parser as character
                 data, regardless of what other
                 delimiters are present */
scandata  ::= /* characters scanned and returned to parser
                 as normal */
anything  ::= /* characters scanned only for ']]' || '>' and not
                 returned to parser.  */
```

39.8 Differences from ISO 8879

This grammar assumes the reference concrete syntax; if an alternate concrete syntax is used, some literal strings given in the DTD and document-instance grammars would need to be replaced accordingly.

White space, entity reference, entity end, and comments within markup declarations are assumed to be handled by the lexical scanner and are omitted from this grammar. This shortens and simplifies the grammar somewhat.

The grammar is written as a Backus-Naur-Form (BNF) grammar rather than a regular-right-part grammar; some additional constructs have thus been introduced to deal with repeating and optional items in the original grammar. Non-terminals optional in the original may be required here, and vice versa, depending on how the optionality of a construct has been expressed; in no case does such a change actually affect the set of strings accepted by the grammar.

Some constructs are omitted entirely because they do not occur in the subset of SGML prescribed for use in the TEI Interchange Format:

- link type declaration
- short reference set
- ranked elements and ranked groups
- data tag group
- minimized start-tags
- formal identifiers

Finally, the recognition and expansion of entity references and the handling of marked sections, CDATA and RCDATA elements, and CDATA, SDATA, or NDATA entities have been ignored in the current version. Entity references are assumed to be handled by the lexical scanner, though in a fully conformant SGML parser they are in part dependent on the state of the syntactic parser. CDATA, RCDATA, SDATA, and NDATA elements or entities, like marked sections, are assumed either not to occur or to be handled by the lexical scanner.

Bibliography

This bibliography lists the works cited in the text of the Guidelines, as well as including some useful publications relating to SGML. It does not include works cited only as sources of examples.

[1] AAP (Association of American Publishers). *Author's Guide to Electronic Manuscript Preparation and Markup.* Version 2.0 Revised Edition. [Dublin, Ohio]: Electronic Publishing Special Interest Group (EPSIG), 1989.

[2] AAP (Association of American Publishers). *Markup of Mathematical Formulas.* Version 2.0 Revised Edition. [Dublin, Ohio]: EPSIG, 1989.

[3] AAP (Association of American Publishers). *Markup of Tabular Material.* Version 2.0 Revised Edition. [Dublin, Ohio]: EPSIG, 1989.

[4] AAP (Association of American Publishers). *Reference Manual on Electronic Manuscript Preparation and Markup.* Version 2.0 Revised Edition. [Dublin, Ohio]: EPSIG, 1989.

[5] ALA (American Library Association). *ALA-LC Romanization Tables: Transliteration Schemes for Non-Roman Scripts*, approved by the Library of Congress and the American Library Association, tables compiled and edited by Randall K. Barry. Washington: Library of Congress, 1991.

[6] Amsler, R. A., and F. W. Tompa, *An SGML-based Standard for English Monolingual Dictionaries. Information in Text, Proc. 4th Conf. of Univ. of Waterloo Centre for the New OED* (Waterloo: [n.p.], October 1988), pp. 61-79.

[7] ANSI (American National Standards Institute). *ANSI X3.4-1986. American National Standard for Information Systems — Coded Character Sets — 7-bit American National Standard Code for Information Interchange (7-bit ASCII).* [New York]: ANSI, 1986.

[8] ANSI (American National Standards Institute). *ANSI X3.41-1974. American National Standard Code Extension Techniques for Use with the 7-bit Coded Character Set of American National Standard Code for Information Interchange.* [New York]: ANSI, 1974.

[9] ANSI (American National Standards Institute). *ANSI Z39.47-1985. American National Standard for Information Sciences — Extended Latin Alphabet Coded Character Set for Bibliographic Use.* [New York]: ANSI, 1984.

[10] Atkinson, J. Maxwell, and John Heritage, eds. *Transcript notation. Structures of social action: Studies in conversation analysis.* Cambridge: Cambridge University Press, 1984.

[11] Avram, Henriette. *The MARC Pilot Project.* Washington D.C.: Library of Congress, 1968, p. 3.

[12] Barnard, David, et al. *SGML-Based Markup for Literary Texts. Computers and the Humanities* 22 (1988): 265-76.

[13] Barron, David *Why use SGML? Electronic Publishing Origination, Dissemination and Design* 2.1 (April 1989): 3-24.

[14] Berkowitz, Luci, and Karl A. Squitier. *Thesaurus Linguae Graecae Canon of Greek Authors and Works*, 2nd edition. Oxford: Oxford University Press, 1986.

[15] Boase, S. *London-Lund Corpus: Example Text and Transcription Guide.* London: Survey of English Usage, University College London, 1990.

[16] Bryan, Martin *SGML: an Author's Guide to the Standard Generalized Markup Language.* Reading: Addison-Wesley, 1988.

[17] Butcher, Judith. *Copy-editing: The Cambridge Handbook.* 2nd edition. Cambridge: Cambridge University Press, 1981.

[18] Byrne, Deborah J. *MARC Manual: Understanding and Using MARC Records.* Englewood, Colo.: Libraries Unlimited, Inc., 1991.

[19] Calzolari, Nicoletta, et al. *Computational Model of the Dictionary Entry: Preliminary Report,* Acquilex: Esprit Basic Research Action No. 3030, Six-Month Deliverable. Pisa, April 1990.

[20] Chartrand, Gary, and Linda Lesniak. *Graphs and Digraphs.* Menlo Park, CA: Wadsworth, 1986).

[21] *The Chicago Manual of Style, for Authors, Editors and Copywriters.* 13th Edition. Chicago and London: Univ. of Chicago Press, 1982.

[22] Coombs, James H., Allen H. Renear, and Steven J. DeRose. *Markup Systems and the Future of Scholarly Text Processing. Communications of the ACM* 30.11 (November 1987): 933-947.

[23] Cover, Robin C., et al. *A Bibliography on Structured Text: Technical Report 90-281* , Queen's University, Kingston, Ont. June 1990

[24] Crawford, Walt. *MARC for Library Use: Understanding the USMARC Formats.* Professional Librarian Series. White Plains, NY: Knowledge Industry Publications, 1984.

[25] The DANLEX Group. *Descriptive Tools for Electronic Processing of Dictionary Data.* In *Lexicographica, Series Maior.* Tübingen: Niemeyer, 1987.

[26] ECMA (European Computer Manufacturers Association). *Standard ECMA-94: 8-Bit Single-Byte Coded Graphic Character Sets. Latin Alphabets No. 1 to No. 4.* 2nd Edition — June 1986. [Geneva]: ECMA, 1986.

[27] ECMA (European Computer Manufacturers Association). *Standard ECMA-113: 8-Bit Single-Byte Coded Graphic Character Sets. Latin/Cyrillic Alphabet.* 2nd Edition — July 1988. [Geneva]: ECMA, 1988.

[28] ECMA (European Computer Manufacturers Association). *Standard ECMA-114: 8-Bit Single-Byte Coded Graphic Character Sets. Latin/Arabic Alphabet.* June 1986. [Geneva]: ECMA, 1986.

[29] ECMA (European Computer Manufacturers Association). *Standard ECMA-118: 8-Bit Single-Byte Coded Graphic Character Sets. Latin/Greek Alphabet.* December 1986. [Geneva]: ECMA, 1986.

[30] ECMA (European Computer Manufacturers Association). *Standard ECMA-121: 8-Bit Single-Byte Coded Graphic Character Sets. Latin/Hebrew Alphabet.* July 1987. [Geneva]: ECMA, 1987.

[31] ECMA (European Computer Manufacturers Association). *Standard ECMA-128: 8-Bit Single-Byte Coded Graphic Character Sets. Latin Alphabet No. 5.* July 1988. [Geneva]: ECMA, 1988.

[32] Edwards, J. A., and M. D. Lampert, eds. *Talking Language: Transcription and Coding of Spoken Discourse* Hillsdale, N.J.: Lawrence Erlbaum Associates, 1993.

[33] Fought, John, and Carol Van Ess-Dykema. *Toward an SGML Document Type Definition for Bilingual Dictionaries.* TEI working paper TEI AIW20 (available from the TEI).

[34] Francis, W. N., and H. Kučera. *Manual of Information to Accompany a Standard Sample of Present-day Edited American English, for Use with Digital Computers.* Original ed. 1964, revised 1971, revised and augmented 1979. Providence, R.I.: Department of Linguistics, Brown University.

[35] Francis, W. Nelson, and Henry Kucera, with the assistance of Andrew W. Mackie. *Frequency Analysis of English Usage: Lexicon and Grammar.* Boston: Houghton Mifflin, 1982.

[36] Gale, William A., and Kenneth W. Church. *Program for Aligning Sentences in Bilingual Corpora. Computational Linguistics* 19 (1993): 75-102.

[37] Garside, R. G., G. N. Leech, and G. R. Sampson. *The Computational Analysis of English: a Corpus-Based Approach.* Oxford: Oxford University Press, 1987.

[38] Gavioli, Laura, and Gillian Mansfield. *The Pixi Corpora.* Bologna: Cooperativa Libraria Universitaria Editrice, 1990.

[39] Geens, D., L.K. Engels, and W. Martin. *Leuven Drama Corpus and Frequency List.* Leuven: University of Leuven, Institute of Applied Linguistics, 1975.

[40] Goldfarb, Charles F., *The SGML Handbook.* Oxford: Clarendon Press, 1990.

[41] Gorman, Michael, and Paul W. Winkler, eds. *Anglo-American Cataloguing Rules.* Second Edition. Chicago: American Library Association; London: Library Association; Ottawa: Canadian Library Association, 1978.

[42] Gorman, Michael. *After AACR2R: The Future of the Anglo-American Cataloging Rules.* In *Origins, Content and Future of AACR2 Revised,* ed. Richard Smiraglia. Chicago: American Library

Association, 1992.

[43] Guittet, C., ed., *Formex: Formalized Exchange of Electronic Publications.* Luxembourg: Office for Official Publications of the European Communities, New Technologies – Project Management Department, 1984.

[44] van Herwijnen, Eric. *Practical SGML.* , Kluwer 1990

[45] Holt, Brian P. *UNIMARC Manual.* London, U.K.: IFLA Universal Bibliographic Control and International MARC Programme, British Library, 1987.

[46] IBM (International Business Machines). *3174 Establishment Controller: Character Set Reference.* Manual no. GA27-3831-01. Second Edition (May 1989). [n.p.]: IBM, 1989.

[47] Ide, Nancy, and Jean Veronis. *Encoding Print Dictionaries. Computers and the Humanities* (1994, in press).

[48] Ide, Nancy, Jacques Le Maitre, and Jean Veronis. *Outline of a Model for Lexical Databases, Information Processing and Management* 29.2 (1993): 159-186.

[49] Ide, Nancy, Jean Veronis, Susan Warwick-Armstrong, and Nicoletta Calzolari. *Principles for Encoding machine readable dictionaries. Proceedings of the Fifth EURALEX International Congress, EURALEX'92* (University of Tempere, Finland). In press.

[50] ISO (International Organization for Standardization). *ISO 639: 1988. Code for the representation of names of languages.* [Geneva]: International Organization for Standardization, 1988.

[51] ISO (International Organization for Standardization). *ISO 646: 1983. Information processing — ISO 7-bit coded character set for information interchange.* [Geneva]: International Organization for Standardization, 1983.

[52] ISO (International Organization for Standardization). *ISO 646: 1991. Information processing — ISO 7-bit coded character set for information interchange.* ([Geneva]: International Organization for Standardization, 1991.)

[53] ISO (International Organization for Standardization). *ISO 2014: 1976. Writing of Calendar Dates in All-Numeric Form.* [Geneva]: International Organization for Standardization, 1976.

[54] ISO (International Organization for Standardization). *ISO 2022: 1986. Information processing — ISO 7-bit and 8-bit coded character sets — Code extension techniques.* 3d ed. [Geneva]: International Organization for Standardization, 1986.

[55] ISO (International Organization for Standardization). *ISO 6429: 1992. Information processing — Control functions for 7-bit and 8-bit coded character sets*. [Geneva]: International Organization for Standardization, 1992.

[56] ISO (International Organization for Standardization). *ISO 8601: 1988. Data elements and interchange formats — Information interchange — Representation of dates and times.* [Geneva]: International Organization for Standardization, 1988.

[57] ISO (International Organization for Standardization). *ISO 8859-1: 1987 (E). Information processing — 8-bit Single-Byte Coded Graphic Character Sets — Part 1: Latin Alphabet No. 1.* (*Traitement de l'information — Jeux de caracte"res graphiques code"s sur un seul octet — Partie 1: Alphabet latin no 1.*) First edition — 1987-02-15. [Geneva]: International Organization for Standardization, 1987.

[58] ISO (International Organization for Standardization). *ISO 8859-2: 1987 (E). Information processing — 8-bit Single-Byte Coded Graphic Character Sets — Part 2: Latin Alphabet No. 2.* (*Traitement de l'information — Jeux de caracte"res graphiques code"s sur un seul octet — Partie 2: Alphabet latin no 2.*) First edition — 1987-02-15. [Geneva]: International Organization for Standardization, 1987.

[59] ISO (International Organization for Standardization). *ISO 8859-7: 1987 (E). Information processing — 8-bit Single-Byte Coded Graphic Character Sets — Part 7: Latin/Greek Alphabet.* First edition — 1987-11-15. [Geneva]: International Organization for Standardization, 1987.

[60] ISO (International Organization for Standardization). *ISO 8859-8: 1988 (E). Information processing — 8-bit Single-Byte Coded Graphic Character Sets — Part 8: Latin/Hebrew Alphabet.* (*Traitement de l'information — Jeux de caracte"res graphiques code"s sur un seul octet — Partie 8: Alphabet Latin/He"breu.*) First edition — 1988-06-01. [Geneva]: International Organization for Standardization, 1988.

[61] ISO (International Organization for Standardization). *ISO 8879-1986 (E). Information processing — Text and Office Systems — Standard Generalized Markup Language (SGML).* First edition

— 1986-10-15. [Geneva]: International Organization for Standardization, 1986.

[62] ISO (International Organization for Standardization). *ISO 8879:1986 / A1:1988 (E). Information processing — Text and Office Systems — Standard Generalized Markup Language (SGML), Amendment 1*. Published 1988-07-01. [Geneva]: International Organization for Standardization, 1988.

[63] ISO (International Organization for Standardization). *ISO/TR 9573-1988(E). Information processing—SGML support facilities—Techniques for using SGML.* Final text of 1988-09-12.

[64]

[65] ISO (International Organization for Standardization). *ISO 10241: 1993. Preparation and layout of international terminology standards.* [Geneva]: International Organization for Standardization, 1993.

[66] ISO (International Organization for Standardization), and IEC (International Electrotechnical Commission). *ISO/IEC 10646-1: 1993. Information technology — Universal Multiple-Octet Coded Character Set (UCS) — Part 1: Architecture and Basic Multilingual Plane.* [Geneva]: International Organization for Standardization, 1993.

[67]

[68] ISO (International Organization for Standardization), and IEC (International Electrotechnical Commission). *ISO/IEC 10744: 1992. Information Technology — Hypermedia/Time-based Structuring Language (HyTime).* [Geneva]: International Organization for Standardization, 1992.

[69] Jackendoff, R. *X-Bar Syntax: A Study in Phrase Structure.* Linguistic Inquiry Monographs, 2. Cambridge: MIT Press, 1977.

[70] Johansson, Stig. *Encoding a Corpus in Machine-Readable Form*. In *Computational Approaches to the Lexicon: An Overview,* ed. B. T. S. Atkins et al. Oxford: Oxford University Press, forthcoming.

[71] Johansson, Stig, in collaboration with Eric Atwell, Roger Garside, and Geoffrey Leech. *The Tagged LOB Corpus: User's Manual*. Bergen: Norwegian Computing Centre for the Humanities, 1986.

[72] Johansson, S., G. Leech, and H. Goodluck. *Manual of Information to Accompany the Lancaster-Oslo/Bergen Corpus of British English, for Use with Digital Computers.* Oslo: Department of English, University of Oslo, 1978.

[73] Johansson, Stig, et al. *Working Paper on Spoken Texts,* document TEI AI2 W1, 1991.

[74] Kucera, Henry, and W. Nelson Francis. *Computational analysis of present-day American English*, with a foreword by W. F. Twaddell, a study by Mary Lois Marckworth and Laura M. Bell, and an analytical essay by John B. Carroll. Providence, Brown University Press, 1967.

[75] Kytö, M., and M. Rissanen. *The Helsinki Corpus of English Texts,* in *Corpus Linguistics: Hard and Soft*, ed. M. Kytö, O. Ihalainen, and M. Rissanen. Amsterdam: Rodopi, 1988.

[76] Langendoen, D. Terence, and Gary F. Simons. *A Rationale for the TEI Recommendations for Feature-Structure Markup. Computers and the Humanities* (1994; in press).

[77] Leech, G. N., and R. G. Garside. *Running a Grammar Factory,* in *English Computer Corpora: Selected Papers and Research Guide,* ed. S. Johansson and A.-B. Stenström. Berlin: de Gruyter; New York: Mouton, 1991, pp. 15-32.

[78] Loman, Bengt, and Nils Jørgensen. *Manual for analys och beskrivning av macrosyntagmer.* Lundastudier i nordisk språkvetenskap, serie C nr. 1. Lund: Studentlitteratur, 1971.

[79] MacWhinney, Brian. *CHAT Manual.* [Pittsburgh]: Dept of Psychology, Carnegie-Mellon University, 1988, pp. 87ff.

[80] Marshall, I. *Choice of Grammatical Word Class without Global Syntactic Analysis: Tagging Words in the LOB Corpus. Computers and the Humanities* 17 (1983): 139-50.

[81] Modern Humanities Research Association. *MHRA style book: notes for authors, editors and writers of dissertations.* 3rd edition. London: MHRA, 1981.

[82] Pereira, Fernando C. N. *Grammars and Logics of Partial Information.* SRI International Technical Note 420. Menlo Park, CA: SRI International, 1987.

[83] Renouf, A. *Corpus Development.* In *Looking Up: An Account of the COBUILD Project in Lexical Computing.* Ed. J. McH. Sinclair. London: Collins ELT, 1987.

[84] Shieber, Stuart. *An Introduction to Unification-based Approaches to Grammar.* CSLI Lecture Notes 4. Palo Alto, CA: Center for the Study of Language and Information, 1986.

[85] *Sociolinguistics/Soziolinguistik (An international handbook of the science of language and society. Ein internationales Handbuch zur Wissenschaft von Sprache und Gesellschaft)*. Berlin, New York: De Gruyter, 1988. I, pp. 271 and 274.

[86] Stokes, Roy, rev. *Esdaile's Manual of Bibliography,* revised edition by Roy Stokes. London: George Allen and Unwin Ltd, 1931; 1967.

[87] TEI (Text Encoding Initiative). *Design Principles for Text Encoding Guidelines.* Document No. TEI EDP1. Version 2, 10 January 1990.

[88] TEI. *Charges to the Working Committees.* Document No. TEI EDP2. Version 3, 28 August 1989.

[89] *TEI / LISA / ISO - TIF — Terminology Interchange Format — A Tutorial.* Rpt. in *TermNet News* no 40 (1993), pp. 5-64.

[90] Warmer, J., and S. van Egmond *The implementation of the Amsterdam SGML parser. Electronic Publishing Origination, Dissemination and Design* 2.2 (July 1989): 65-90.

[91] TLG (Thesaurus Linguae Graecae). *Beta Manual.* (Irvine: TLG, [1988]).

[92] The Unicode Consortium. *The Unicode Standard: Worldwide Character Encoding.* Version 1.0, Volume 1. Reading, Mass.: Addison-Wesley, 1991.

[93] *USMARC format for bibliographic data, including content designation*. Washington, D.C.: Library of Congress, 1987.

[94] *Webster's New Collegiate Dictionary.* Springfield, Mass.: G. & C. Merriam Co., 1975. (Cited as: Webster's Collegiate.)

[95] *Webster's New International Dictionary of the English Language*, ed. W. T. Harris and F. Sturges Allen. Springfield, Mass.: G. & C. Merriam Co., 1922. (Cited as: Webster's Second International.)

[96] Wu, Gilbert S.K. *SGML theory and practice.* 1989 *British Library research paper*68

Index

Colophon

The text of this manual was prepared electronically on a variety of systems. Most sections were originally drafted by members of the work groups and working committees of the TEI; all have been revised by the editors to achieve greater uniformity of style and greater consistency in the tag set. SGML tags from a preliminary version of the tag set documented here were introduced in the text by the original drafters of the sections or by the editors, using standard text editors on VMS, VM/CMS, PC-DOS, and Apple Macintosh systems. The resulting SGML document fragments were validated using VM2 (a markup validator distributed with the public-domain ARCSGML materials), sgmls (a parser built on the same engine), Markit, Checkmark, XGML Validator, and/or Author/Editor. The validated files, which contained both the prose description of the tag set and the reference materials, were then processed with ad hoc programs written in Spitbol to produce separate SGML documents for the prose, the reference section, and the DTDs; these SGML documents were then further translated by simple programs in Rexx and Spitbol into either LaTeX or Waterloo GML for formatting and printing. The camera ready copy of this document was produced by Waterloo Script GML, version 89.1, running on the IBM 3090 at the University of Illinois at Chicago Academic Computer Center (to which thanks), and using a set of macros extending the GML Starter Set of tags with specialized tags designed for this document.