TEI Lite: An Introduction to Text Encoding for Interchange

Lou Burnard
C. M. Sperberg-McQueen
June 1995

1. Prefatory note

TEI Lite was the name adopted for what the TEI editors originally conceived of as a simple demonstration of how the TEI encoding scheme might be adopted to meet 90% of the needs of 90% of the TEI user community. In retrospect, it was predictable that many people should imagine TEI Lite to be all there is to TEI, or find TEI Lite to be far too heavy for their needs (to meet the latter criticism, Michael also prepared a special barebones version of TEI Lite).

TEI Lite was based largely on our observations of existing and previous practice in the encoding of texts, particularly as manifest in the collections of the Oxford Text Archive and in our own experience. It is therefore unsurprising that it seems to have become, if not a de facto standard, at least a common point of departure for electronic text centres and encoding projects world wide. Maybe the fact that we actually produced this shortish, readable, manual for it also helped.

That manual was, of course, authored and is maintained in the DTD it describes, originally as an SGML document, more recently in XML. This made it easy to produce a number of differently formatted versions in HTML, Tex, PDF, etc., some of which can be found in The TEI Vault. As well as the HTML version you are reading, the document is also available for download as a rather nicely formatted PDF document

Early adopters of TEI Lite included a number of ‘Electronic Text Centers’, many of whom produced their own documentation and tutorial materials (some examples are listed in the TEI Tutorials pages). The present tutorial has also been translated into a gratifyingly large number of languages other than English. Here’s the list as of October 1999: if you know of (or would like to produce) a version in another language, please get in touch!

- In French: La TEI simplifiée: une introduction au codage des textes electroniques (translated by François Role; also available in PDF; and in TEI format)
- In Italian: TEI Lite: introduzione alla codifica dei testi (translated by Fabio Ciotti et al)
- In Japanese: here
- in Korean: here (only in SGML)
- in Russian: here (only in RTF)

Although there is also now an XML version of the TEI Lite DTD, the documentation has not yet been changed to reflect this: some examples and issues discussed therefore still relate only to the SGML version.
LouBurnard, June 2000

This document provides an introduction to the recommendations of the Text Encoding Initiative (TEI), by describing a manageable subset of the full TEI encoding scheme. The scheme documented here can be used to encode a wide variety of commonly encountered textual features, in such a way as to maximize the usability of electronic transcriptions and to facilitate their interchange among scholars using different computer systems. It is also fully compatible with the full TEI scheme, as defined by TEI document P3, *Guidelines for Electronic Text Encoding and Interchange*, published in Chicago and Oxford in May 1994.

1. Introduction

The Text Encoding Initiative (TEI) Guidelines are addressed to anyone who wants to interchange information stored in an electronic form. They emphasize the interchange of textual information, but other forms of information such as images and sound are also addressed. The Guidelines are equally applicable in the creation of new resources and in the interchange of existing ones.

The Guidelines provide a means of making explicit certain features of a text in such a way as to aid the processing of that text by computer programs running on different machines. This process of making explicit we call *markup* or *encoding*. Any textual representation on a computer uses some form of markup; the TEI came into being partly because of the enormous variety of mutually incomprehensible encoding schemes currently besetting scholarship, and partly because of the expanding range of scholarly uses now being identified for texts in electronic form.

The TEI Guidelines use the *Standard Generalized Markup Language* (SGML) to define their encoding scheme. SGML is an international standard (ISO 8879), used increasingly throughout the information processing industries, which makes possible a formal definition of an encoding scheme, in terms of *elements* and *attributes*, and rules governing their appearance within a text. The TEI's use of SGML is ambitious in its complexity and generality, but it is fundamentally no different from that of any other SGML markup scheme, and so any general-purpose SGML-aware software is able to process TEI-conformant texts.

The TEI is sponsored by the Association for Computers and the Humanities, the Association for Computational Linguistics, and the Association for Literary and Linguistic Computing. Funding has been provided in part from the U.S. National Endowment for the Humanities, 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. Its Guidelines were published in May 1994, after six years of development involving many hundreds of scholars from different academic disciplines worldwide.

At the outset of its work, the overall goals of the TEI were defined by the closing statement of a planning conference held at Vassar College, N.Y., in November, 1987; these 'Poughkeepsie Principles' were further elaborated in a series of design documents. The Guidelines, say these design documents, should:

- suffice to represent the textual features needed for research;
- be simple, clear, and concrete;
- be easy for researchers to use without special-purpose software;
- allow the rigorous definition and efficient processing of texts;
- provide for user-defined extensions;
- conform to existing and emergent standards.

The world of scholarship is large and diverse. For the Guidelines to have wide acceptability, it was important to ensure that:

1. the common core of textual features be easily shared;
2. additional specialist features be easy to add to (or remove from) a text;

3. multiple parallel encodings of the same feature should be possible;
4. the richness of markup should be user-defined, with a very small minimal requirement;
5. adequate documentation of the text and its encoding should be provided.

The present document describes a manageable selection from the extensive set of SGML elements and recommendations resulting from those design goals, which is called *TEI Lite*.

In selecting from the several hundred SGML elements defined by the full TEI scheme, we have tried to identify a useful ‘starter set’, comprising the elements which almost every user should know about. Experience working with TEI Lite will be invaluable in understanding the full TEI DTD and in knowing which optional parts of the full DTD are necessary for work with particular types of text.

Our goals in defining this subset may be summarized as follows:

- it should include most of the TEI ‘core’ tag set, since this contains elements relevant to virtually all text types and all kinds of text-processing work;
- it should be able to handle adequately a reasonably wide variety of texts, at the level of detail found in existing practice (as demonstrated in, for example, the holdings of the Oxford Text Archive);
- it should be useful for the production of new documents as well as encoding of existing ones;
- it should be usable with a wide range of existing SGML software;
- it should be derivable from the full TEI DTD using the extension mechanisms described in the TEI Guidelines;
- it should be as small and simple as is consistent with the other goals.

The reader may judge our success in meeting these goals for him or herself. At the time of writing, our confidence that we have at least partially done so is borne out by its use in practice for the encoding of real texts. The Oxford Text Archive uses TEI Lite when it translates texts from its holdings from their original markup schemes into SGML; the Electronic Text Centers at the University of Virginia and the University of Michigan have used TEI Lite to encode their holdings. And the Text Encoding Initiative itself uses TEI Lite, in its current technical documentation --- including this document.

Although we have tried to make this document self-contained, as suits a tutorial text, the reader should be aware that it does not cover every detail of the TEI encoding scheme. All of the elements described here are fully documented in the TEI Guidelines themselves, which should be consulted for authoritative reference information on these, and on the many others which are not described here. Some basic knowledge of SGML is assumed.

2. A Short Example

We begin with a short example, intended to show what happens when a passage of prose is typed into a computer by someone with little sense of the purpose of mark-up, or the potential of electronic texts. In an ideal world, such output might be generated by a very accurate optical scanner. It attempts to be faithful to the appearance of the printed text, by retaining the original line breaks, by introducing blanks to represent the layout of the original headings and page breaks, and so forth. Where characters not available on the keyboard are needed (such as the accented letter *a* in *faàl* or the long dash), it attempts to mimic their appearance.

```
                        CHAPTER 38

READER, I married him. A quiet wedding we had: he and I, the par-
son and clerk, were alone present. When we got back from church, I
went into the kitchen of the manor-house, where Mary was cooking
the dinner, and John cleaning the knives, and I said --
   'Mary, I have been married to Mr Rochester this morning.' The
housekeeper and her husband were of that decent, phlegmatic
order of people, to whom one may at any time safely communicate a
remarkable piece of news without incurring the danger of having
one's ears pierced by some shrill ejaculation and subsequently stunned
by a torrent of wordy wonderment. Mary did look up, and she did
stare at me; the ladle with which she was basting a pair of chickens
roasting at the fire, did for some three minutes hang suspended in air,
and for the same space of time John's knives also had rest from the
polishing process; but Mary, bending again over the roast, said only --
   'Have you, miss? Well, for sure!'
   A short time after she pursued, 'I seed you go out with the master,
but I didn't know you were gone to church to be wed'; and she
basted away. John, when I turned to him, was grinning from ear to
ear.
   'I telled Mary how it would be,' he said: 'I knew what Mr Ed-
ward' (John was an old servant, and had known his master when he
was the cadet of the house, therefore he often gave him his Christian
name) -- 'I knew what Mr Edward would do; and I was certain he
would not wait long either: and he's done right, for aught I know. I
wish you joy, miss!' and he politely pulled his forelock.
   'Thank you, John. Mr Rochester told me to give you and Mary
this.'
   I put into his hand a five-pound note.  Without waiting to hear
more, I left the kitchen. In passing the door of that sanctum some time
after, I caught the words --
   'She'll happen do better for him nor ony o' t' grand ladies.' And
again, 'If she ben't one o' th' handsomest, she's noan faa\l, and varry
good-natured; and i' his een she's fair beautiful, onybody may see
that.'
   I wrote to Moor House and to Cambridge immediately, to say what
I had done: fully explaining also why I had thus acted. Diana and

                              474

                  JANE EYRE                    475

Mary approved the step unreservedly. Diana announced that she
would just give me time to get over the honeymoon, and then she
would come and see me.
   'She had better not wait till then, Jane,' said Mr Rochester, when I
read her letter to him; 'if she does, she will be too late, for our honey-
moon will shine our life long: its beams will only fade over your
grave or mine.'
   How St John received the news I don't know: he never answered
the letter in which I communicated it: yet six months after he wrote
to me, without, however, mentioning Mr Rochester's name or allud-
ing to my marriage. His letter was then calm, and though very serious,
kind. He has maintained a regular, though not very frequent correspond-
ence ever since: he hopes I am happy, and trusts I am not of those who
live without God in the world, and only mind earthly things.
```

This transcription suffers from a number of shortcomings:

- the page numbers and running titles are intermingled with the text in a way which makes it difficult for software to disentangle them;

- no distinction is made between single quotation marks and apostrophe, so it is difficult to know exactly which passages are in direct speech;
- the preservation of the copy text’s hyphenation means that simple-minded search programs will not find the broken words;
- the accented letter in *faàl* and the long dash have been rendered by ad hoc keying conventions which follow no standard pattern and will be processed correctly only if the transcriber remembers to mention them in the documentation;
- paragraph divisions are marked only by the use of white space, and hard carriage returns have been introduced at the end of each line. Consequently, if the size of type used to print the text changes, reformatting will be problematic.

We now present the same passage, as it might be encoded using the TEI Guidelines. As we shall see, there are many ways in which this encoding could be extended, but as a minimum, the TEI approach allows us to represent the following distinctions:

- Paragraph divisions are now marked explicitly.
- Apostrophes are distinguished from quotation marks.
- Entity references are used for the accented letter and the long dash.
- Page divisions have been marked with an empty `<pb>` element alone.
- To simplify searching and processing, the lineation of the original has not been retained and words broken by typographic accident at the end of a line have been re-assembled without comment. If the original lineation were of interest, as it might be for an important printing, it could easily be recorded, though it has not been here.
- For convenience of proof reading, a new line has been introduced at the start of each paragraph, but the indentation is removed.

```
<pb n=’474’/>
<div1 type="chapter" n=’38’>

<p>Reader, I married him.  A quiet wedding we had: he and I,
the parson and clerk, were alone present.  When we got back
from church, I went into the kitchen of the manor-house,
where Mary was cooking the dinner, and John cleaning the
knives, and I said &dash;</p>

<p><q>Mary, I have been married to Mr Rochester this
morning.</q> The housekeeper and her husband were of that
decent, phlegmatic order of people, to whom one may at any
time safely communicate a remarkable piece of news without
incurring the danger of having one’s ears pierced by some
shrill ejaculation and subsequently stunned by a torrent of
wordy wonderment.  Mary did look up, and she did stare at
me; the ladle with which she was basting a pair of chickens
roasting at the fire, did for some three minutes hang
suspended in air, and for the same space of time John’s
knives also had rest from the polishing process; but Mary,
bending again over the roast, said only &dash;</p>

<p><q>Have you, miss? Well, for sure!</q></p>

<p>A short time after she pursued, <q>I seed you go out with
```

```
the master, but I didn’t know you were gone to church to be
wed</q>; and she basted away.  John, when I turned to him,
was grinning from ear to ear.  <q>I telled Mary how it would
be,</q> he said: <q>I knew what Mr Edward</q> (John was an
old servant, and had known his master when he was the cadet
of the house, therefore he often gave him his Christian
name) &dash; <q>I knew what Mr Edward would do; and I was
certain he would not wait long either: and he’s done right,
for aught I know.  I wish you joy, miss!</q> and he politely
pulled his forelock.</p>

<p><q>Thank you, John.  Mr Rochester told me to give you and
Mary this.</q></p>

<p>I put into his hand a five-pound note.  Without waiting
to hear more, I left the kitchen.  In passing the door of
that sanctum some time after, I caught the words &dash;</p>

<p><q>She’ll happen do better for him nor ony o’ t’ grand
ladies.</q> And again, <q>If she ben’t one o’ th’
handsomest, she’s noan fa&agrave;l, and varry good-natured;
and i’ his een she’s fair beautiful, onybody may see
that.</q></p>

<p>I wrote to Moor House and to Cambridge immediately, to
say what I had done: fully explaining also why I had thus
acted.  Diana and <pb n=’475’/> Mary approved the step
unreservedly.  Diana announced that she would just give me
time to get over the honeymoon, and then she would come and
see me.</p>

<p><q>She had better not wait till then, Jane,</q> said Mr
Rochester, when I read her letter to him; <q>if she does,
she will be too late, for our honeymoon will shine our life
long: its beams will only fade over your grave or mine.</q></p>

<p>How St John received the news I don’t know: he never
answered the letter in which I communicated it: yet six
months after he wrote to me, without, however, mentioning Mr
Rochester’s name or alluding to my marriage.  His letter was
then calm, and though very serious, kind.  He has maintained
a regular, though not very frequent correspondence ever
since: he hopes I am happy, and trusts I am not of those who
live without God in the world, and only mind earthly things.</p>
```

The decision to focus on Brontë’s text, rather than on the printing of it in this particular edition, is one aspect of a fundamental encoding issue: that of selectivity. An encoding makes explicit only those textual features of importance to the encoder. It is not difficult to think of ways in which the encoding of even this short passage might readily be extended. For example:

- a regularized form of the passages in dialect could be provided;
- footnotes glossing or commenting on any passage could be added;
- pointers linking parts of this text to others could be added;
- proper names of various kinds could be distinguished from the surrounding text;
- detailed bibliographic information about the text’s provenance and context could be prefixed to it;

- a linguistic analysis of the passage into sentences, clauses, words, etc., could be provided, each unit being associated with appropriate category codes;
- the text could be segmented into narrative or discourse units;
- systematic analysis or interpretation of the text could be included in the encoding, with potentially complex alignment or linkage between the text and the analysis, or between the text and one or more translations of it;
- passages in the text could be linked to images or sound held on other media.

The TEI-recommended way of carrying all of these out is described in the remainder of this document. The TEI scheme as a whole also provides for an enormous range of other possibilities, of which we cite only a few:

- detailed analysis of the components of names;
- detailed meta-information providing thesaurus-style information about the text's origins or topics;
- information about the printing history or manuscript variations exhibited by a particular series of versions of the text.

For recommendations on these and many other possibilities, the full Guidelines should be consulted.

3. The Structure of a TEI Text

All TEI-conformant texts contain (a) a *TEI header* (marked up as a `<teiHeader>` element) and (b) the transcription of the text proper (marked up as a `<text>` element).

The TEI header provides information analogous to that provided by the title page of a printed text. It has up to four parts: a bibliographic description of the machine-readable text, a description of the way it has been encoded, a non-bibliographic description of the text (a *text profile*), and a revision history. The header is described in more detail in section 20 (The Electronic Title Page).

A TEI text may be *unitary* (a single work) or *composite* (a collection of single works, such as an anthology). In either case, the text may have an optional *front* or *back*. In between is the *body* of the text, which, in the case of a composite text, may consist of *group*s, each containing more groups or texts.

A unitary text will be encoded using an overall structure like this:

```
<TEI.2>
    <teiHeader> [ TEI Header information ]  </teiHeader>
    <text>
        <front> [ front matter ... ]   </front>
        <body>  [ body of text ... ]   </body>
        <back>  [ back matter ...  ]   </back>
    </text>
</TEI.2>
```

A composite text also has an optional front and back. In between occur one or more groups of texts, each with its own optional front and back matter. A composite text will thus be encoded using an overall structure like this:

```
<TEI.2>
    <teiHeader> [ header information for the composite ] </teiHeader>
    <text>
        <front> [ front matter for the composite  ]      </front>
        <group>
```

```
           <text>
              <front> [ front matter of first text ] </front>
              <body>  [ body of first text  ]          </body>
              <back>  [ back matter of first text ]    </back>
           </text>
           <text>
              <front> [ front matter of second text]  </front>
              <body>  [ body of second text  ]          </body>
              <back>  [ back matter of second text ]    </back>
           </text>
           [ more texts or groups of texts here ]
        </group>
        <back>      [ back matter for the composite  ]      </back>
    </text>
</TEI.2>
```

It is also possible to define a composite of TEI texts, each with its own header. Such a collection is known as a *TEI corpus*, and may itself have a header:

```
<teiCorpus>
  <teiHeader>   [header information for the corpus]</teiHeader>
  <TEI.2>
     <teiHeader>[header information for first text]</teiHeader>
     <text>     [first text in corpus]             </text>
  </TEI.2>
  <TEI.2>
    <teiHeader>[header information for second text]</teiHeader>
    <text>     [second text in corpus]             </text>
  </TEI.2>
</teiCorpus>
```

It is not however possible to create a composite of corpora -- that is, a number of `<teiCorpus>` elements combined together and treated as a single object. This is a restriction of the current version of the TEI Guidelines.

In the remainder of this document, we discuss chiefly simple text structures. The discussion in each case consists of a short list of relevant TEI *elements* with a brief definition of each, followed by definitions for any *attributes* specific to that element. In most cases, short examples are also given.

4. Encoding the Body

As indicated above, a simple TEI document at the textual level consists of the following elements:

`<front>` contains any prefatory matter (headers, title page, prefaces, dedications, etc.) found before the start of a text proper.

`<group>` contains a number of unitary texts or groups of texts.

`<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.

Elements specific to front and back matter are described below in section 19 (Front and Back Matter). In this section we discuss the elements making up the body of a text.

4.1. Text Division Elements

The body of a prose text may be just a series of paragraphs, or these paragraphs may be grouped together into chapters, sections, subsections, etc. In the former case, each paragraph is tagged using the `<p>` tag. In the latter case, the `<body>` may be divided either into a series of `<div1>` elements, or into a series of `<div>` elements, either of which may be further subdivided, as discussed below:

`<p>`	marks paragraphs in prose.
`<div>`	contains a subdivision of the front, body, or back 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).

When structural subdivisions smaller than a `<div1>` are necessary, a `<div1>` may be divided into `<div2>` elements, a `<div2>` into smaller `<div3>` elements, etc., down to the level of `<div7>`. If more than seven levels of structural division are present, one must either modify the TEI tag set to accept `<div8>`, etc., or else use the unnumbered `<div>` element: a `<div>` may be subdivided by smaller `<div>` elements, without limit to the depth of nesting.

All these *division* elements take the following three attributes:

type	This indicates the conventional name for this category of text division. Its value will typically be “Book”, “Chapter”, “Poem”, etc. Other possible values include “Group” for groups of poems, etc., treated as a single unit, “Sonnet”, “Speech”, and “Song”. Note that whatever value is supplied for the type attribute of the first `<div>`, `<div1>`, `<div2>`, etc., in a text is assumed to apply for all subsequent `<div>`, `<div1>`s (etc.) within the same `<body>`. This implies that a value must be given for the first division element of each type, or whenever the value changes.
id	This specifies a unique identifier for the division, which may be used for cross references or other links to it, such as a commentary, as further discussed in section 8 (Cross References and Links). It is often useful to provide an id attribute for every major structural unit in a text, and to derive the ID values in some systematic way, for example by appending a section number to a short code for the title of the work in question, as in the examples below.
n	The n attribute specifies a mnemonic short name or number for the division, which can be used to identify it in preference to the ID. If a conventional form of reference or abbreviation for the parts of a work already exists (such as the book/chapter/verse pattern of Biblical citations), the n attribute is the place to record it.

The attributes id and n, indeed, are so widely useful that they are allowed on any element in any TEI DTD: they are *global attributes*. Other global attributes defined in the TEI Lite scheme are discussed in section 8.3 (Linking Attributes).

The value of every id attribute must be unique within a document. One simple way of ensuring that this is so is to make it reflect the hierarchic structure of the document. For example, Smith’s *Wealth of Nations* as first published consists of five books, each of which is divided into chapters, while some chapters are further subdivided into parts. We might define id values for this structure as follows:

```
<div1 id="WN1" n='I' type='book'>
  <div2 id="WN101" n='I.1' type='chapter'>
```

```
     ... </div2>
    <div2 id="WN102" n='I.2' type='chapter'>
     ... </div2>
     ...
    <div2 id="WN110" n='I.10' type='chapter'>
       <div3 id="WN1101" n='I.10.1' type="part">
        ... </div3>
       <div3 id="WN1102" n='I.10.2' type="part">
        ... </div3>
    </div2>
    ...
   </div1>
   <div1 id="WN2" n='II' type='book'>
     ....
   </div1>
  ...
```

A different numbering scheme may be used for id and n attributes: this is often useful where a canonical reference scheme is used which does not tally with the structure of the work. For example, in a novel divided into books each containing chapters, where the chapters are numbered sequentially through the whole work, rather than within each book, one might use a scheme such as the following:

```
<div1 id="TS01" n='1' type='Volume'>
   <div2 id="TS011" n='1' type='Chapter'>
      ... </div2>
   <div2 id="TS012" n='2'>
      ...</div2>
 </div1>
 <div1 id="TS02" n='2' type='Volume'>
   <div2 id="TS021" n='3'type='Chapter'>
      ...</div2>
   <div2 id="TS022" n='4'>
      ...</div2>
</div1>
```

Here the work has two volumes, each containing two chapters. The chapters are numbered conventionally 1 to 4, but the id values specified allow them to be regarded additionally as if they were numbered 1.1, 1.2, 2.1, 2.2.

4.2. Headings and Closings

Every `<div>`, `<div1>`, `<div2>`, etc., may have a title or heading at its start, and (less commonly) a closing such as “End of Chapter 1”. The following elements may be used to transcribe them:

`<head>` contains any heading, for example, the title of a section, or the heading of a list or glossary.

`<trailer>` contains a closing title or footer appearing at the end of a division of a text.

Some other elements which may be necessary at the beginning or ending of text divisions are discussed below in section 19.1.2 (Prefatory Matter) .

Whether or not headings and trailers are included in a transcription is a matter for the individual transcriber to decide. Where a heading is completely regular (for example “Chapter 1”) or has been given as an attribute value (e.g. `<div1 type='Chapter' n=1>`), it may be omitted; where it contains otherwise unrecoverable text it should always be included. For example, the start of Hardy’s *Under the Greenwood Tree* might be encoded as follows:

```
<div1 id="UGT1" n='Winter' type='Part'>
<div2 id="UGT11" n='1' type='Chapter'>
<head>Mellstock-Lane</head>
<p>To dwellers in a wood almost every species of tree ...
```

4.3. *Prose, Verse and Drama*

As noted above, the paragraphs making up a textual division should be tagged with the `<p>` tag. For example:

```
<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.
</p>
</body>
```

A number of different tags are provided for the encoding of the structural components of verse and performance texts (drama, film, etc.):

`<l>` contains a single, possibly incomplete, line of verse. Attributes include:

part specifies whether or not the line is metrically complete. Legal values are: `F` for the final part of an incomplete line, `Y` if the line is metrically incomplete, `N` if the line is complete, or if no claim is made as to its completeness, `I` for the initial part of an incomplete line, `M` for a medial 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.

`<sp>` contains 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>` contains a special form of heading or label, giving the name of one or more speakers in a performance text or fragment.

`<stage>` contains any kind of stage direction within a performance text or fragment. Attributes include:

type indicates the kind of stage direction. Suggested values include *entrance*, *exit*, *setting*, *delivery*, etc.

Here, for example, is the start of a poetic text in which verse lines and stanzas are tagged:

```
<lg n="I">
<l>I Sing the progresse of a
   deathlesse soule,</l>
<l>Whom Fate, with God made,
  but doth not controule,</l>
<l>Plac'd in most shapes; all times
  before the law</l>
<l>Yoak'd us, and when, and since,
  in this I sing.</l>
<l>And the great world to his aged evening;</l>
<l>From infant morne, through manly noone I draw.</l>
<l>What the gold Chaldee, of silver Persian saw,</l>
<l>Greeke brass, or Roman iron, is in this one;</l>
<l>A worke t'out weare Seths pillars, bricke and stone,</l>
```

```
<l>And (holy writs excepted) made to yeeld to none,</l>
</lg>
```

Note that the `<l>` element marks verse, not typographic lines: the original lineation of the first few lines above has not therefore been made explicit by this encoding, and may be lost. The `<lb>` element described in section 5 (Page and Line Numbers) may be used to mark typographic lines if so desired.

Sometimes, particularly in dramatic texts, verse lines are split between speakers. The easiest way of encoding this is to use the part attribute to indicate that the lines so fragmented are incomplete, as in this example:

```
<div1 type ='Act' n='I'><head>ACT I</head>
<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?</l></speaker></sp>
<sp><speaker>Fran<l>Nay, answer me. Stand and unfold
  yourself.</l></speaker></sp>
<sp><speaker>Barn<l part="i">Long live the King!</l></speaker></sp>
<sp><speaker>Fran<l part="m">Barnardo?</l></speaker></sp>
<sp><speaker>Barn<l part="f">He.</l></speaker></sp>
<sp><speaker>Fran<l>You come most carefully upon
  your hour.</l></speaker></sp>
```

The same mechanism may be applied to stanzas which are divided between two speakers:

```
<sp><speaker>First voice</speaker>
<lg type="stanza" part="I">
<l>But why drives on that ship so fast</l>
<l>Withouten wave or wind?</l>
</lg>
<sp><speaker>Second Voice</speaker>
<lg part="F">
<l>The air is cut away before.</l>
<l>And closes from behind.</l>
</lg>
```

This example shows how dialogue presented in a prose work as if it were drama should be encoded. It also demonstrates the use of the who attribute to bear a code identifying the speaker of the piece of dialogue concerned:

```
<sp who="OPI"><speaker>The reverend Doctor Opimiam</speaker>
<p>I do not think I have named a single unpresentable fish.
<sp who="GRM"><speaker>Mr Gryll</speaker>
<p>Bream, Doctor: there is not much to be said for bream.</p>
<sp who="OPI"><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>
<p>Fish, Miss Gryll -- I could discourse to you on fish by
the hour:  but for the present I will forbear...</p>
</sp>
```

5. Page and Line Numbers

Page and line breaks may be marked with the following empty elements.

`<pb>` marks the boundary between one page of a text and the next in a standard reference system.

`<lb>` marks the start of a new (typographic) line in some edition or version of a text.

These elements mark a single point in the text, not a span of text. The global n attribute should be used to supply the number of the page or line beginning at the tag. In addition, these two elements share the following attribute:

ed	indicates the edition or version in which the page break is located at this point.

When working from a paginated original, it is often useful to record its pagination, if only to simplify later proof-reading. Recording the line breaks may be useful for the same reason; treatment of end-of-line hyphenation in printed source texts will require some consideration.

If pagination, etc., are marked for more than one edition, specify the edition in question using the ed attribute, and supply as many tags are necessary. For example, in the following passage we indicate where the page breaks occur in two different editions (ED1 and ED2)

```
<p>I wrote to Moor House and to Cambridge immediately, to
say what I had done: fully explaining also why I had thus
acted.  Diana and <pb ed="ED1" n='475'/> Mary approved the
step unreservedly.  Diana announced that she would
<pb ed="ED2" n='485'/>just give me time to get over the
honeymoon, and then she would come and see me.</p>
```

The `<pb>` and `<lb>` elements are special cases of the general class of *milestone* elements which mark reference points within a text. TEI Lite also includes a generic `<milestone>` element, which is not restricted to special cases but can mark any kind of reference point: for example, a column break, the start of a new kind of section not otherwise tagged, etc. This element has the following description and attributes:

`<milestone>` marks the boundary between sections of a text, as indicated by changes in a standard reference system. Attributes include:

ed	indicates the edition or version to which the milestone applies.
unit	indicates what kind of section is changing at this milestone.

The names used for types of unit and for editions referred to by the ed and unit attributes may be chosen freely, but should be documented in the header.

The `<milestone>` element may be used to replace the others, or the others may be used as a set; they should not be mixed arbitrarily.

6. Marking Highlighted Phrases

6.1. Changes of Typeface, etc.

Highlighted words or phrases are those made visibly different from the rest of the text, typically by a change of type font, handwriting style, or ink color, intended to draw the reader's attention to them.

The global rend attribute can be attached to any element, and used wherever necessary to specify details of the highlighting used for it. For example, a heading rendered in bold might be tagged `head rend='Bold'`, and one in italic `head rend='Italic'`.

It is not always possible or desirable to interpret the reasons for such changes of rendering in a text. In such cases, the element `<hi>` may be used to mark a sequence of highlighted text without making any claim as to its status.

`<hi>` marks a word or phrase as graphically distinct from the surrounding text, for reasons concerning which no claim is made.

In the following example, the use of a distinct typeface for the subheading and for the included name are recorded but not interpreted:

```
<p><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 ...</p>
```

Alternatively, where the cause for the highlighting can be identified with confidence, a number of other, more specific, elements are available.

`<emph>` marks words or phrases which are stressed or emphasized for linguistic or rhetorical effect.

`<foreign>` identifies a word or phrase as belonging to some language other than that of the surrounding text.

`<mentioned>` marks words or phrases mentioned, not used.

`<term>` contains a single-word, multi-word or symbolic designation which is regarded as a technical term.

`<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: *m* for monographic title (book, collection, or other item published as a distinct item, including single volumes of multi-volume works); *s* (series title); *j* (journal title); *u* for title of unpublished material (including theses and dissertations unless published by a commercial press); *a* for analytic title (article, poem, or other item published as part of a larger item).

type classifies the title according to some convenient typology. Sample values include: `abbreviated`, `main`, `subordinate` (for subtitles and titles of parts), and `parallel` (for alternate titles, often in another language, by which the work is also known).

Some features (notably quotations and glosses) may be found in a text either marked by highlighting, or with quotation marks. In either case, the elements `<q>` and `<gloss>` (as discussed in the following section) should be used. If the rendition is to be recorded, use the global rend attribute.

As an example of the elements defined 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.

Interpreting the role of the highlighting, the sentence might look like this:

```
<p>On the one hand the <title>Nibelungenlied</title> is associated
with the new rise of romance of twelfth-century France, the
<foreign>romans d’antiquit&eacute;</foreign>, the romances of
Chr&eacute;tien de Troyes, ...</p>
```

Describing only the appearance of the original, it might look like this:

```
<p>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">romans
d’antiquit&eacute;</hi>, the romances of
Chr&eacute;tien de Troyes, ...</p>
```

6.2. Quotations and Related Features

Like changes of typeface, quotation marks are conventionally used to denote several different features within a text, of which the most frequent is quotation. When possible, we recommend that the underlying feature be tagged, rather than the simple fact that quotation marks appear in the text, using the following elements:

- `<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 a character or speaker; in dictionaries, `<q>` may be used to mark real or contrived examples of usage. Attributes include:
  - **type** may be used to indicate whether the quoted matter is spoken or thought, or to characterize it more finely. Sample values include: *spoken* (for representation of direct speech, usually marked by quotation marks) and *thought* (for representation of thought, e.g. internal monologue).
  - **who** identifies the speaker of a piece of direct speech.
- `<mentioned>` marks words or phrases mentioned, not used.
- `<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.
- `<gloss>` marks a word or phrase which provides a gloss or definition for some other word or phrase. Attributes include:
  - **target** identifies the associated word or phrase.

Here is a simple example of a quotation:

```
<p>Few dictionary makers are likely to forget
Dr. Johnson's description of the
lexicographer as <q>a harmless drudge.</q></p>
```

To record how a quotation was printed (for example, *in-line* or set off as a *display* or *block quotation*), the rend attribute should be used. This may also be used to indicate the kind of quotation marks used.

Direct speech interrupted by a narrator can be represented simply by ending the quotation and beginning it again after the interruption, as in the following example:

```
<p><q>Who-e debel you?</q> &mdash; he at last said &mdash; <q>you
no speak-e, damme, I kill-e.</q>  And so saying, the lighted
tomahawk began flourishing about me in the dark.</p>
```

If it is important to convey the idea that the two `<q>` elements together reproduce a single speech, the linking attributes next and prev may be used, as described in section 8.3 (Linking Attributes).

Quotations may be accompanied by a reference to the source or speaker, using the who attribute, whether or not the source is given in the text, as in the following example:

```
<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>
```

This example also demonstrates how quotations may be embedded within other quotations: one speaker (Wilson) quotes another speaker (Spaulding).

The creator of the electronic text must decide whether quotation marks are replaced by the tags or whether the tags are added and the quotation marks kept. If the quotation marks are removed from the text, the rend attribute may be used to record the way in which they were rendered in the copy text.

As with highlighting, it is not always possible and may not be considered desirable to interpret the function of quotation marks in a text in this way. In such cases, the tag `<hi rend=quoted>` might be used to mark quoted text without making any claim as to its status.

6.3. *Foreign Words or Expressions*

Words or phrases which are not in the main language of the texts may be tagged as such in one of two ways. If the word or phrase is already tagged for some reason, the element indicated should bear a value for the global lang attribute indicating the language used. Where there is no applicable element, the element `<foreign>` may be used, again using the lang attribute. For example:

```
<p>John has real <foreign lang="fra">savoir-faire</foreign>.</p>
<p>Have you read <title lang="deu">Die Dreigroschenoper</title>?</p>
<p><mentioned lang="fra">Savoir-faire</mentioned> is French for know-how.</p>
<p>The court issued a writ of <term lang="lat">mandamus</term>.</p>
```

As these examples show, the `<foreign>` element should not be used to tag foreign words if some other more specific element such as `<title>`, `<mentioned>`, or `<term>` applies. The global lang attribute may be attached to any element to show that it uses some other language than that of the surrounding text.

7. Notes

All notes, whether printed as footnotes, endnotes, marginalia, or elsewhere, should be marked using the same element:

`<note>` contains a note or annotation. Attributes include:

- **type** describes the type of note.
- **resp** indicates who is responsible for the annotation: author, editor, translator, etc. The value might be `author`, `editor`, etc., or the initials of the individual who added the annotation.
- **place** indicates where the note appears in the source text. Sample values include `inline`, `interlinear`, `left`, `right`, `foot`, and `end`, for notes which appear as marked paragraphs in the body of the text, between the lines, in the left or right margin, at the foot of the page, or at the end of the chapter or volume, respectively.
- **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.
- **anchored** indicates whether the copy text shows the exact place of reference for the note.

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 marginalia, 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. Notes may also be placed in a separate division of the text (as end-notes are, in printed books) and linked to the relevant portion of the text using their target attribute.

The n attribute may be used to supply the number or identifier of a note if this is required. The resp attribute should be used consistently to distinguish between authorial and editorial notes, if the work has both kinds; otherwise, the TEI header should state which kind they are.

Examples:

```
<p>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 ...</p>
```

```
<lg id="RAM609">
<note place="margin">The curse is finally expiated</note>
<l>And now this spell was snapt: once more</l>
<l>I viewed the ocean green,</l>
<l>And looked far forth, yet little saw</l>
<l>Of what had else been seen &dash;</l>
</lg>
```

8. Cross References and Links

Explicit cross references or links from one point in a text to another in the same SGML document may be encoded using the elements described in section 8.1 (Simple Cross References). References or links to elements of some other SGML document, or to parts of non-SGML documents, may be encoded using the *TEI extended pointers* described in section 8.2 (Extended Pointers). Implicit links (such as the association between two parallel texts, or that between a text and its interpretation) may be encoded using the linking attributes discussed in section 8.3 (Linking Attributes).

8.1. Simple Cross References

A cross reference from one point within a single document to another can be encoded using either of the following elements:

`<ref>` 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>` a pointer to another location in the current document in terms of one or more identifiable elements.

These elements share the following attributes:

target specifies the destination of the pointer as one or more SGML identifiers

type categorizes the pointer in some respect, using any convenient set of categories.

targType specifies the type (or types) of element to which this pointer may point.

crDate specifies when this pointer was made.

resp specifies the creator of the pointer.

The difference between these two elements is that `<ptr>` is an empty element, simply marking a point from which a link is to be made, whereas `<ref>` may contain some text as well --- typically the text of the cross-reference itself. The `<ptr>` element would be used for a cross reference which is to be indicated by some non-verbal means such as a symbol or icon, or in an electronic text by a button. It is also useful in document production systems, where the formatter can generate the correct verbal form of the cross reference.

The following two forms, for example, are logically equivalent (assuming we have documented somewhere the exact verbal form of cross references represented by `<ptr>` elements):

```
See especially <ref target="SEC12">section 12 on page 34</ref>.
```

```
See especially <ptr target="SEC12"/>.
```

The value of the target attribute must be an SGML identifier in the current SGML document. This implies that the passage or phrase being pointed at must bear an identifier, and must therefore be tagged as an element of some kind. In the following example, the cross reference is to a `<div1>` element:

```
...
see especially <ptr target="SEC12"/>.
...
<div1 id="SEC12"><head>Concerning Identifiers...
...
```

Because the id attribute is global, any element in a document may be pointed to in this way. In the following example, a paragraph has been given an identifier so that it may be pointed at:

```
...
this is discussed in <ref target="pspec">the paragraph on links</ref>
...
<p id="pspec">Links may be made to any kind of element
...
```

The targType attribute can be used to specify that the element pointed to must be of a particular type, as in the following example:

```
...
this is discussed in <ref target="dspec" targType='div1 div2'>
the section on links</ref>
```

This reference should fail if the element with identifier dspec is not either a `<div1>` or a `<div2>`. Note however that this check cannot be carried out by an SGML parser alone, since the SGML parser can only check that some element dspec exists.

The type attribute can be used to categorize the link represented by the pointer in any convenient way. The resp and crDate attributes may also be used to represent the person or agency responsible for making the link, and its date of creation, as in the following example:

```
 ...
this is discussed in
<ref type="xref" resp="auto" crdate="950521" target="dspec" targType='div1 div2'>
the section on links</ref>
```

These attributes are most likely to be of use in hypertext systems containing very many pointers used for a variety of purposes and created by a variety of means.

Sometimes the target of a cross reference does not correspond with any particular feature of a text, and so may not be tagged as an element of some kind. If the desired target is simply a point in the current document, the easiest way to mark it is by introducing an `<anchor>` element at the appropriate spot. If the target is some sequence of words not otherwise tagged, the `<seg>` element may be introduced to mark them. These two elements are described as follows:

`<anchor>` specifies a location or point within a document so that it may be pointed to.

`<seg>` identifies a span or segment of text within a document so that it may be pointed to. Attributes include

type categorizes the segment

In the following (imaginary) example, `<ref>` elements have been used to represent points in this text which are to be linked in some way to other parts of it; in the first case to a point, and in the second, to a sequence of words:

```
Returning to <ref target="ABCD">the point where I dozed
off</ref>, I noticed that <ref target="EFGH">three
words</ref> had been circled in red by a previous reader
```

This encoding requires that elements with the specified identifiers (ABCD and EFGH in this example) are to be found somewhere else in the current document. Assuming that no element already exists to carry these identifiers, the `<anchor>` and `<seg>` elements may be used:

```
.... <anchor type="bookmark" id='ABCD'/> ....
 ....<seg type="target" id='EFGH'> ... </seg> ...
```

The type attribute should be used (as above) to distinguish amongst different purposes for which these general purpose elements might be used in a text. Some other uses are discussed in section 8.3 (Linking Attributes) below.

8.2. Extended Pointers

The elements `<ptr>` and `<ref>` can only be used for cross-references or links whose targets occur within the same SGML document as their source. They can also refer only to SGML elements. The elements discussed in this section are not restricted in this way.

`<xptr>` defines a pointer to another location in the current document or an external document.

`<xref>` defines a pointer to another location in the current document or an external document, possibly modified by additional text or comment.

In addition to the pointer attributes already discussed in section 8.1 (Simple Cross References) above, these elements share the following additional attributes, which are used to specify the target of the cross reference or link in place of the target attribute:

doc specifies the document within which the required location is to be found, by default the current document.

from specifies the start of the destination of the pointer as an expression in the TEI extended pointer syntax, by default the whole of the document indicated by the doc attribute.

to specifies the endpoint of the destination of the pointer as an expression in the TEI extended pointer syntax; may only be specified if the from attribute has been.

A full specification of the language used to express the target of TEI extended pointers is beyond the scope of this document; here we list here only a few of its more generally useful features. The full Guidelines should be consulted for more detail.

An `<xptr>` (or `<xref>`) may point to the whole of some other document simply by supplying an entity name as the value of the doc attribute, as in this example:

```
see <xref doc="P3">The TEI Guidelines, passim</xref>
```

This example assumes that some system or public entity with the name P3 has been declared. This declaration may be placed within the litemods.ent extension file or in some other manner specific to the particular SGML authoring software in use (as discussed in section 15 (Figures and Graphics)).

The from attribute is used to specify some location within whatever document is specified by the doc attribute. The specification uses a special language, called the *TEI extended pointer syntax*; only some details of which are given here. In this language, locations are defined as a series of *steps*, each one identifying some part of the document, often in terms of the locations identified by the previous step. For example, you would point to the third sentence of the second paragraph of chapter two by selecting chapter two in the first step, the second paragraph in the second step, and the third sentence in the last step. A step can be defined in terms of SGML concepts (such as *parent*, *descendent*, *preceding*, etc.) or, more loosely, in terms of text patterns, word or character positions. You can also use a foreign (non-SGML) notation, or specify a location within a graphic in terms of its co-ordinate system.

The from and to attributes use the same notation. Each points to some portion of the target document; the extended pointer as a whole points to the section beginning at the start of the from and running to the end of the to.

The first step in a location path will often be to specify the identifier of some element within the target document, as in this example:

```
<xptr doc="P3" from='id (SA)'/>
```

This selects the whole of whatever element bears the identifier SA within the entity P3. If a finer-grained target is required, other steps might follow. The following keywords are available for you to select other elements in terms of their relationship to this one:

child elements contained by this one.

ancestor elements which contains this one, directly or indirectly.

previous elements with the same parent as this one but preceding it in the document.

next elements with the same parent as this one and following it in the document.

preceding elements in the document which start before this one does, irrespective of their parents.

following elements in the document which start after this one does, irrespective of their parents.

Each of these keywords implies a particular set of elements (the set of children, the set of ancestors, the set of previous siblings, etc.); to specify which element in the set we are pointing at, the keyword may optionally be followed by a parenthesized list containing:

- a positive or negative number, indicating which of the possibly many elements found is intended (+1 indicating the first element encountered, starting from the current location, and -1 indicating the last), or the keyword *all*, indicating that all the elements in the set are to be pointed at;

- a generic identifier, indicating the type of element required, or a star indicating that any element type will do;
- a set of attribute names and values, indicating that the element selected should have attributes with the names and values specified, if any.

Continuing the above example, the following reference will select the third `<p>` element directly contained by whatever element has the identifier SA:

```
<xptr doc="P3" from='id (SA) child (3 p)'/>
```

Similarly, assuming that the entity P3 is in fact a reference to the SGML form of the TEI Guidelines, then the following reference will select section 14.2.2 of that publication in which (as it happens) the extended pointer syntax is formally defined:

```
For full details, see
<ref doc="P3" from='id (SA) child (2 div2) child (2 div3)'>
  TEI Extended pointer syntax definition
</ref>
```

Normally, the scope of a cross reference will be adequately defined by the from attribute. For some documents, however, it may be more convenient to define both a starting and an ending scope. As noted above, the to attribute is provided for this purpose. For example,

```
<xptr doc="P1" from='id (xyz)' to='id (abc)'/>
```

is an extended pointer whose target is the sequence starting at the beginning of whatever element in document P1 has identifier XYZ and ending at the end of whatever element in the same document has identifier ABC. Any elements in between are also included, irrespective of structure; the pointer is erroneous if the end of ABC precedes the start of XYZ.

Very complex specifications are easily built using this syntax. For example, the following reference will select the most recent `<head>` element which carries an attribute lang with the value *LAT*, and which occurs before the start of the element with identifier SA:

```
<xptr doc="P3" from='id (SA) preceding (1 head lang lat)'/>
```

If no value is supplied for the doc attribute, the current document is assumed. Thus, the following references are semantically equivalent. They both indicate the element with identifier X1 within the current document:

```
<ptr target="X1"/>
<xptr from='id (X1)'/>
```

8.3. Linking Attributes

The following special purpose *linking* attributes are defined for every element in the TEI Lite DTD:

ana	links an element with its interpretation.
corresp	links an element with one or more other corresponding elements.
next	links an element to the next element in an aggregate.
prev	links an element to the previous element in an aggregate.

The ana (analysis) attribute is intended for use where a set of abstract analyses or interpretations have been defined somewhere within a document, as further discussed in section 16 (Interpretation and Analysis). For example, a linguistic analysis of the sentence “John loves Nancy” might be encoded as follows:

```
<seg type="sentence" ana="SVO">
  <seg type="lex" ana="NP1">John</seg>
  <seg type="lex" ana="VVI">loves</seg>
  <seg type="lex" ana="NP1">Nancy</seg>
</seg>
```

This encoding implies the existence elsewhere in the document of elements with identifiers SVO, NP1, and VV1 where the significance of these particular codes is explained. Note the use of the `<seg>` element to mark particular components of the analysis, distinguished by the type attribute.

The corresp (corresponding) attribute provides a simple way of representing some form of correspondence between two elements in a text. For example, in a multilingual text, it may be used to link translation equivalents, as in the following example

```
<seg lang="FRA" id="FR1" corresp="EN1">Jean aime Nancy</seg>
<seg lang="ENG" id="EN1" corresp="FR1">John loves Nancy</seg>
```

The same mechanism may be used for a variety of purposes. In the following example, it has been used to represent anaphoric correspondences between “the show” and “Shirley”, and between “NBC” and “the network”:

```
<p><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.</p>
```

The next and prev attributes provide a simple way of linking together the components of a discontinuous element, as in the following example:

```
<q id="Q1a" next="Q1b">Who-e debel you?</q>
&mdash he at last said &mdash
<q id="Q1b" prev="Q1a">you no speak-e,
damme, I kill-e.</q>  And so saying,
the lighted tomahawk began flourishing
about me in the dark.
```

9. Editorial Interventions

The process of encoding an electronic text has much in common with the process of editing a manuscript or other text for printed publication. In both cases a conscientious editor may wish to record both the original state of the source and any editorial correction or other change made in it. The elements discussed in this and the next section provide some facilities for meeting these needs.

The following pair of elements may be used to mark *correction*, that is editorial changes introduced where the editor believes the original to be erroneous:

- `<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.
- `<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.
  - **cert** signifies the degree of certainty ascribed to the correction.

The following pair of elements may be used to mark *normalization*, that is editorial changes introduced for the sake of consistency or modernization of a text:

- `<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.
- `<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.

For example, the reading

```
... for his nose was as sharp as a pen and a’ table of green feelds
```

is taken by Gifford as involving (1) the erroneous substitution of *table* for *babbled*, and (2) the non-standard spellings *a’* and *feelds* for *he* and *fields*. Gifford’s conjecture might be encoded thus:

```
... for his nose was as sharp as a pen and <reg sic="a’">he</reg>
 <corr sic=’table’ ed="Gifford">babbl’d</corr> of green
<reg sic=’feelds’>fields</reg>
```

10. Omissions, Deletions, and Additions

In addition to correcting or normalizing words and phrases, editors and transcribers may also supply missing material, omit material, or transcribe material deleted or crossed out in the source. In addition, some material may be particularly hard to transcribe because it is hard to make out on the page. The following elements may be used to record such phenomena:

- `<add>` contains letters, words, or phrases inserted in the text by an author, scribe, annotator, or corrector. Attributes include:
  - **place** if the addition is written into the copy text, indicates where the additional text is written. Sample values include `inline`, `supralinear`, `infralinear`, `left` (in left margin), `right` (in right margin), `top`, `bottom`, etc.
- `<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.
  - **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.

`<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.

hand signifies the hand of the agent which made the deletion.

`<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.

These elements may be used to record changes made by an editor, by the transcriber, or (in manuscript material) by the author or scribe. For example, if the source for an electronic text read

```
The following elements are provided for
for simple editorial interventions.
```

then it might be felt desirable to correct the obvious error, but at the same time to record the deletion of the superfluous second *for*, thus:

```
The following elements are provided for
<del hand="LB">for</del> simple editorial interventions.
```

The attribute value `LB` on the hand attribute indicates that “LB” corrected the duplication of *for*.

If the source read

```
The following elements provided for
for simple editorial interventions.
```

(i.e. if the verb had been inadvertently dropped) then the corrected text might read:

```
The following elements <add hand="LB">are</add> provided for
<del hand="LB">for</del> simple editorial interventions.
```

The attribute value `LB` on the hand attribute indicates that “LB” corrected the duplication of *for*.

These elements are not limited to changes made by an editor; they can also be used to record authorial changes in manuscripts. A manuscript in which the author has first written “How it galls me, what a galling shadow”, then crossed out the word *galls* and inserted *dogs* might be encoded thus:

```
How it <del hand="DHL" type="overstrike">galls</del>
<add hand="DHL" place="supralinear">dogs</add> me,
what a galling shadow
```

Similarly, the `<unclear>` and `<gap>` elements may be used together to indicate the omission of illegible material; the following example also shows the use of `<add>` for a conjectural emendation:

```
One hundred & twenty good regulars joined to me
<unclear><gap reason='indecipherable'/></unclear>
& instantly, would aid me signally <add hand="ed">in?</add>
an enterprise against Wilmington.
```

The `<del>` element marks material which is transcribed as part of the electronic text despite being marked as deleted, while `<gap>` marks the location of material which is omitted from the electronic text, whether it is legible or not. A language corpus, for example, might omit long quotations in foreign languages:

```
<p> ... An example of a list appearing in a fief ledger of
<name type="place">Koldinghus</name> <date>1611/12</date>
is given below. It shows cash income from a sale of
honey.</p>
<q><gap desc='quotation from ledger'
    reason='in Danish'/></q>
<p>A description of the overall structure of the account is
once again ... </p>
```

Other corpora (particular those constructed before the widespread use of scanners) systematically omit figures and mathematics:

```
<p>At the bottom of your screen below the mode line is the
<term>minibuffer</term>.  This is the area where Emacs
echoes the commands you enter and where you specify
filenames for Emacs to find, values for search and replace,
and so on.
<gap desc='diagram of Emacs screen' reason='graphic'/>
</p>
```

11. Names, Dates, Numbers and Abbreviations

The TEI scheme defines elements for a large number of ‘data-like’ features which may appear almost anywhere within almost any kind of text. These features may be of particular interest in a range of disciplines; they all relate to objects external to the text itself, such as the names of persons and places, numbers and dates. They also pose particular problems for many natural language processing (NLP) applications because of the variety of ways in which they may be presented within a text. The elements described here, by making such features explicit, reduce the complexity of processing texts containing them.

11.1. Names and 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.

The type attribute is used to distinguish amongst (for example) names of persons, places and organizations, where this is possible:

```
<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>
```

```
It being one of the principles of the
<rs type="organization">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 necessarily one 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.

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 are also available for these and similar elements 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.

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:

```
<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.
```

More detailed tagging of the components of proper names is also possible, using the additional tag set for names and dates.

11.2. Dates and Times

Tags for the more detailed encoding of times and dates include the following:

`<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.

The value attribute specifies a normalized form for the date or time, using a recognized format such as ISO 8601. Partial dates or times (e.g. “1990”, “September 1990”, “twelvish”) can usually be expressed by simply omitting a part of the value supplied; alternatively 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.

Examples:

```
<date value='1980-02-21'>21 Feb 1980</date>
<date value='1990'>1990</date>
<date value='1990-09'>September 1990</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>
```

```
<l>specially when it's nine below zero</l>
<l>and <time value='15:00'>three o'clock in the afternoon</time></l>
```

11.3. Numbers

Numbers 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*). In natural-language processing or machine-translation applications, it is often helpful to distinguish them from other, more ‘lexical’ parts of the text. In other applications, the ability to record a number’s value in standard notation is important. The `<num>` element provides this possibility:

`<num>` contains a number, written in any form. Attributes include:

type indicates the type of numeric value. Suggested values include: *fraction*, *ordinal* (for ordinal numbers, e.g. “21st”), *percentage*, and *cardinal* (an absolute number, e.g. “21”, “21.5”, etc.)

value supplies the value of the number in an application-dependent standard form.

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>
```

11.4. Abbreviations and their Expansion

Like names, dates, and numbers, abbreviations may be transcribed as they stand or expanded; they may be left unmarked, or encoded using the following element:

`<abbr>` contains an abbreviation of any sort. Attributes include:

expan gives an expansion of the abbreviation.

type allows the encoder to classify the abbreviation according to some convenient typology. Sample values include *contraction*, *suspension*, *brevigraph*, *superscription*, or *acronym*. The type attribute may also be given values like *title* (for titles of address), *geographic*, *organization*, etc., describing the nature of the object referred to.

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:

```
<name><abbr type="title" expan='Doctor'>Dr.</abbr>
<abbr type="initial" expan='Marilyn'>M.</abbr>
Deegan</name>
is the Director of the
<abbr expan='Computers in Teaching Initiative' type="acronym">
CTI</abbr> Centre for Textual Studies.
```

This element is also particularly useful where manuscript materials in which abbreviation is very frequent are being transcribed.

11.5. Addresses

The `<address>` element is used to mark a postal address of any kind. It contains one or more `<addrLine>` elements, one for each line of the address.

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.

Here is a simple example:

```
<address>
<addrLine>Computer Center (M/C 135)</addrLine>
<addrLine>1940 W. Taylor, Room 124</addrLine>
<addrLine>Chicago, IL 60612-7352</addrLine>
<addrLine>U.S.A.</addrLine>
</address>
```

The individual parts of an address may be further distinguished by using the `<name>` element discussed above (section 11.1 (Names and Referring Strings)).

```
<address>
<addrLine>Computer Center (M/C 135)</addrLine>
<addrLine>1940 W. Taylor, Room 124</addrLine>
<addrLine><name type="city">Chicago</name>, IL 60612-7352</addrLine>
<addrLine><name type="country">USA</name></addrLine>
</address>
```

12. Lists

The element `<list>` is used to mark any kind of *list*. A list is a sequence of text items, which may be ordered, unordered, or a glossary list. Each item may be preceded by an item label (in a glossary list, this label is the term being defined):

`<list>` contains any sequence of items organized as a list. Attributes include:

- **type** describes the form of the list. Suggested values include: *ordered*, *bulleted* (for lists with numbered or lettered items, and lists with bullet-marked items, respectively), *gloss* (for lists consisting of a set of technical terms, each marked with a `<label>` element and accompanied by a gloss or definition marked as an `<item>`), and *simple* (for lists with items not marked with number or bullets.

`<item>` contains one component of a list.

`<label>` contains the label associated with an item in a list; in glossaries, marks the term being defined.

Individual list items are tagged with `<item>`. The first `<item>` may optionally be preceded by a `<head>`, which gives a heading for the list. The numbering of a list may be omitted (if reconstructible), indicated using the `n` attribute on each item, or (rarely) tagged as content using the `<label>` element. The following are all thus equivalent:

```
<list>
<head>A short list</head>
<item>First item in list.</item>
<item>Second item in list.</item>
<item>Third item in list.</item>
</list>

<list>
<head>A short list</head>
<item n=1>First item in list.</item>
<item n=2>Second item in list.</item>
<item n=3>Third item in list.</item>
</list>

<list>
<head>A short list</head>
<label>1</label><item>First item in list.</item>
<label>2</label><item>Second item in list.</item>
<label>3</label><item>Third item in list.</item>
</list>
```

The styles should not be mixed in the same list.

A simple two-column table may be treated as a *glossary list*, tagged `<list type=gloss>`. Here, each item comprises a *term* and a *gloss*, marked with `<label>` and `<item>` respectively. These correspond to the elements `<term>` and `<gloss>`, which can occur anywhere in prose text.

```
<list type="gloss">
<head>Vocabulary</head>
<label lang="enm">nu</label>        <item>now</item>
<label lang="enm">lhude</label>     <item>loudly</item>
<label lang="enm">bloweth</label>   <item>blooms</item>
<label lang="enm">med</label>       <item>meadow</item>
<label lang="enm">wude</label>      <item>wood</item>
<label lang="enm">awe</label>       <item>ewe</item>
<label lang="enm">lhouth</label>    <item>lows</item>
<label lang="enm">sterteth</label>  <item>bounds, frisks</item>
<label lang="enm">verteth</label>   <item lang="lat">pedit</item>
<label lang="enm">murie</label>     <item>merrily</item>
<label lang="enm">swik</label>      <item>cease</item>
<label lang="enm">naver</label>     <item>never</item>
</list>
```

Where the internal structure of a list item is more complex, it may be preferable to regard the list as a *table*, for which special-purpose tagging is defined in an additional TEI tag set.

Lists of whatever kind can, of course, nest within list items to any depth required. Here, for 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 -->
```

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 rend="run-on"><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>
```

Lists of bibliographic items should be tagged using the `<listBibl>` element, described in the next section.

13. Bibliographic Citations

It is often useful to distinguish bibliographic citations where they occur within texts being transcribed for research, if only so that they will be properly formatted when the text is printed out. The element `<bibl>` is provided for this purpose:

`<bibl>` contains a loosely-structured bibliographic citation of which the sub-components may or may not be explicitly tagged.

Where the components of a bibliographic reference are to be distinguished, the following elements may be used as appropriate. It is generally useful to mark at least those parts (such as the titles of articles, books, and journals) which will need special formatting. The other elements are provided for cases where particular interest attaches to such details.

`<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.

`<biblScope>` defines the scope of a bibliographic reference, for example as a list of page numbers, or a named subdivision of a larger work.

`<date>` contains a date in any format.

`<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. Attributes include:

role specifies the nature of the intellectual responsibility. Sample values include *translator*, *compiler*, *illustrator*, etc.; the default value is *editor*.

`<imprint>` groups information relating to the publication or distribution of a bibliographic item.

`<publisher>` provides the name of the organization responsible for the publication or distribution of a bibliographic item.

`<pubPlace>` contains the name of the place where a bibliographic item was published.

`<series>` contains information about the series in which a book or other bibliographic item has appeared.

`<title>` contains the title of a work, whether article, book, journal, or series, including any alternative titles or subtitles. Attributes include

type categorizes the title in some way, for example as a *main*, *subordinate*, etc.

level indicates the bibliographic *level* or class of title. Legal values are described in section 6.1 (Changes of Typeface, etc.)

For example, the following editorial note might be transcribed as shown:

He was a member of Parliament for Warwickshire in 1445, and died March 14, 1470 (according to Kittredge, *Harvard Studies* 5. 88ff).

```
He was a member of Parliament for Warwickshire in 1445, and died
March 14, 1470 (according to <bibl><author>Kittredge</author>,
<title>Harvard Studies</title> <biblScope>5. 88ff</biblScope></bibl>).
```

For lists of bibliographic citations, the `<listBibl>` element should be used; it may contain a series of `<bibl>` elements. For an example, see the list in Appendix B (References).

14. Tables

Tables represent a sizable challenge for any text processing system, but simple tables, at least, appear in so many texts that even in the simplified TEI tag set presented here, markup for tables is necessary. The following elements are provided for this purpose:

`<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* for labels or descriptive information, and *data* for actual 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* for labels or descriptive information, and *data* for actual data values.

cols indicates the number of columns occupied by this cell.

rows indicates the number of rows occupied by this cell.

For example, Defoe uses mortality tables like the following in the *Journal of the Plague Year* to show the rise and ebb of the epidemic:

```
<p>It was indeed coming on amain, for the burials that
same week were in the next adjoining parishes thus:&mdash;
<table rows="5" cols="4">
<row role=’data’>
<cell role=’label’>St. Leonard’s, Shoreditch</cell>
      <cell>64</cell> <cell>84</cell> <cell>119</cell></row>
<cell role=’label’>St. Botolph’s, Bishopsgate</row>
      <cell>65</cell> <cell>105</cell> <cell>116</cell></row>
<cell role=’label’>St. Giles’s, Cripplegate</row>
     <cell>213</cell> <cell>421</cell> <cell>554</cell></row>
</table>
<p>This shutting up of houses was at first counted a very cruel
and unchristian method, and the poor people so confined made
bitter lamentations. ... </p>
```

15. Figures and Graphics

Not all the components of a document are necessarily textual. The most straightforward text will often contain diagrams or illustrations, to say nothing of documents in which image and text are inextricably intertwined, or electronic resources in which the two are complementary.

The encoder may simply record the presence of a graphic within the text, possibly with a brief description of its content, by using the elements described in this section. The same elements may also be used to embed digitized versions of the graphic within an electronic document.

`<figure>` marks the spot at which a graphic is to be inserted in a document. Attributes include:

entity the name of a pre-defined system entity containing a digitized version of the graphic to be inserted.

`<figDesc>` contains a textual description of the appearance or content of a graphic, for use when documenting an image without displaying it.

Any textual information accompanying the graphic, such as a heading and/or caption, may be included within the `<figure>` element itself, in a `<head>` and one or more `<p>` elements, as may also any text appearing within the graphic itself. It is strongly recommended that a prose description of the image be supplied, as the content of a `<figDesc>` element, for the use of applications which are not able to render the graphic, and to render the document accessible to vision-impaired readers. (Such text is not normally considered part of the document proper.)

The simplest use for these elements is to mark the position of a graphic, as in this example;

```
<pb n="412"/>
<figure></figure>
<pb n="413"/>
```

(Note that the end-tag may not be omitted, even though the element has no content). More usually, a graphic will have at the least an identifying title, which should be encoded using the `<head>` element. It is also often convenient to include a brief description of the image, as in the following example:

```
<figure>
  <head>Mr Fezziwig's Ball</head>
  <figDesc>A Cruikshank engraving showing Mr Fezziwig leading
     a group of revellers.</figDesc>
</figure>
```

When a digitized version of the graphic concerned is available, it is clearly preferable to embed it at the appropriate point within the document. Graphic elements such as pictures are typically stored in separate entities (files) from those containing the text of a document, and using a different notation (storage format). The TEI Lite DTD supports graphics encoded using the CGM, TIFF, or JPEG standards under the SGML notation names *cgm*, *tiff*, and *jpeg*. (Other notations may however be used, provided that an appropriate NOTATION declaration is added to the DTD; see the chapter on tables, formulae, and graphics in TEI P3, or any reference work on SGML, for more details of the SGML *NOTATION* declaration.)

Whatever format is used to encode the image, it may be embedded within the document in the same way. The first step is to declare an SGML entity of a particular type, which specifies a name for the entity, an external identifier (such as a file name) for it, and the notation used. For example, assuming that the digitized image of Mr Fezziwig's ball were held in TIFF format in the file fezzi.tff, an entity declaration like the following would be necessary:

```
<!ENTITY fezziPic SYSTEM "fezzi.tff" NDATA tiff>
```

All such declarations must be processed before the SGML document itself; with the TEI Lite DTD this may be accomplished by including them in a file called litedecls.ent or whatever file has the public identifier -//TEI U5-1995//DTD TEI Lite 1.0 Extensions//EN.

With the above declaration in force, all that is necessary to embed the digitized image at the appropriate point in the document is to supply a value for the entity attribute of the `<figure>` element:

```
<figure entity="fezziPic">
 <head>Mr Fezziwig's Ball</head>
 <figDesc>A Cruikshank engraving showing Mr Fezziwig leading
    a group of revellers.</figDesc>
 </figure>
```

16. Interpretation and Analysis

It is often said that *all* markup is a form of interpretation or analysis. While it is certainly difficult, and may be impossible, to distinguish firmly between 'objective' and 'subjective' information in any universal way, it remains true that judgments concerning the latter are typically regarded as more likely to provide controversy than those concerning the former. Many scholars therefore prefer to record such interpretations only if it is possible to alert the reader that they are considered more open to dispute, than the rest of the markup. This section describes some of the elements provided by the TEI scheme to meet this need.

16.1. Orthographic Sentences

Interpretation typically ranges across the whole of a text, with no particular respect to other structural units. A useful preliminary to intensive interpretation is therefore to segment the text into discrete and identifiable units, each of which can then bear a label for use as a sort of 'canonical reference'. To facilitate such uses, these units may not cross each other, nor nest within each other. They may conveniently be represented using the following element:

`<s>`	identifies an *s-unit* within a document, for purposes of establishing a simple canonical referencing scheme covering the entire text. Attributes include **type** categorizes the unit (e.g. as *declarative*, *interrogative*, etc.)

As the name suggests, the <s> element is most commonly used (in linguistic applications at least) for marking *orthographic sentences*, that is, units defined by orthographic features such as punctuation. For example, the passage from *Jane Eyre* discussed earlier might be divided into s-units as follows:

```
<pb n='474'/>
<div1 type="chapter" n='38'>
<p><s n="001">Reader, I married him.</s>
<s n="002">A quiet wedding we had:</s>
<s n="003">he and I, the parson and clerk, were alone present.</s>
<s n="004">When we got back from church, I went
into the kitchen of the manor-house, where Mary was cooking the dinner,
and John cleaning the knives, and I said &dash;</s>
<p><q><s n="005">Mary, I have been married to Mr Rochester
this morning.</s></q> ...
```

The end-tags shown above are not strictly necessary, since <s> elements cannot nest: the beginning of one <s> element implies that the previous one has finished. When s-units are tagged as shown above, it is advisable to tag the entire text end-to-end, so that every word in the text being analysed will be contained by exactly one <s> element, whose identifier can then be used to specify a unique reference for it. If the identifiers used are unique within the document, then the id attribute might be used in preference to the n used in the above example.

16.2. General-Purpose Interpretation Elements

A more general purpose segmentation element, the <seg> has already been introduced for use in identifying otherwise unmarked targets of cross references and hypertext links (see section 8 (Cross References and Links)); it identifies some phrase-level portion of text to which the encoder may assign a user-specified type, as well as a unique identifier; it may thus be used to tag textual features for which there is no provision in the published TEI Guidelines.

For example, the Guidelines provide no <apostrophe> element to mark parts of a literary text in which the narrator addresses the reader (or hearer) directly. One approach might be to regard these as instances of the <q> element, distinguished from others by an appropriate value for the who attribute. A possibly simpler, and certainly more general, solution would however be to use the <seg> element as follows:

```
<div1 type="chapter" n='38'>
<p><seg type='apostrophe'>Reader, I married him.</seg>
A quiet wedding we had: ...
```

The type attribute on the <seg> element can take any value, and so can be used to record phrase-level phenomena of any kind; it is good practice to record the values used and their significance in the header.

A <seg> element of one type (unlike the <s> element which it superficially resembles) can be nested within a <seg> element of the same or another type. This enables quite complex structures to be represented; some examples were given in section 8.3 (Linking Attributes) above. However, because it must respect the requirement of SGML that elements be properly nested, and may not cut across each other, it cannot cope with the common requirement to associate an interpretation with arbitrary segments of a text which may completely ignore the document hierarchy. It also requires that the interpretation itself be represented by a single coded value in the type attribute.

Neither restriction applies to the <interp> element, which provides powerful features for the encoding of quite complex interpretive information in a relatively straightforward manner.

<interp> provides for an interpretive annotation which can be linked to a span of text. Attributes include:

value identifies the specific phenomenon being annotated.

resp indicates who is responsible for the interpretation.

type indicates what kind of phenomenon is being noted in the passage. Sample values include `image`, `character`, `theme`, `allusion`, or the name of a particular discourse type whose instances are being identified.

inst points to instances of the analysis or interpretation represented by the current element.

`<interpGrp>` collects together `<interp>` tags.

This elements allows the encoder to specify both the class of an interpretation, and the particular instance of that class which the interpretation involves. Thus, whereas with `<seg>` one can say simply that something is an apostrophe, with `<interp>` one can say that it is an instance (apostrophe) of a larger class (rhetorical figures).

Moreover, `<interp>` is an empty element, which must be linked to the passage to which it applies either by means of the ana attribute discussed in section 8.3 (Linking Attributes) above, or by means of its own inst attribute. This means that any kind of analysis can be represented, with no need to respect the SGML document hierarchy, and also facilitates the grouping of analyses of a particular type together. A special purpose `<interpGrp>` element is provided for the latter purpose.

For example, suppose that you wish to mark such diverse aspects of a text as themes or subject matter, rhetorical figures, and the locations of individual scenes of the narrative. Different portions of our sample passage from *Jane Eyre* for example, might be associated with the rhetorical figures of apostrophe, hyperbole, and metaphor; with subject-matter references to churches, servants, cooking, postal service, and honeymoons; and with scenes located in the church, in the kitchen, and in an unspecified location (drawing room?).

These interpretations could be placed anywhere within the `<text>` element; it is however good practice to put them all in the same place (e.g. a separate section of the front or back matter), as in the following example:

```
<back>
<div1 type='Interpretations'>
<p><interp id='fig-apos'  resp='LB, MSM'
     type='figure of speech' value='apostrophe'/>
<interp id='fig-hyp'   resp='LB, MSM'
     type='figure of speech' value='hyperbole'/>
<!-- ... -->
<interp id='set-church'  resp='LB, MSM'
     type='setting' value='church'/>
<!-- ... -->
<interp id='ref-church'  resp='LB, MSM'
     type='reference' value='church'/>
<interp id='ref-serv'    resp='LB, MSM'
     type='reference' value='servants'/>
<!-- ... -->
</p></div1>
```

The evident redundancy of this encoding can be considerably reduced by using the `<interpGrp>` element to group together all those `<interp>` elements which share common attribute values, as follows:

```
<back>
<div1 type='Interpretations'>
<p>
```

```
<interpGrp type='figure of speech' resp='LB, MSM'>
<interp id='fig-apos' value='apostrophe'/>
<interp id='fig-hyp'  value='hyperbole'/>
<interp id='fig-meta' value='metaphor'/>
<!-- ... -->
</interpGrp>
<interpGrp type='scene-setting' resp='LB, MSM'>
<interp id='set-church'  value='church'/>
<interp id='set-kitch'   value='kitchen'/>
<interp id='set-unspec'  value='unspecified'/>
<!-- ... -->
</interpGrp>
<interpGrp type='reference' resp='LB, MSM'>
<interp id='ref-church' value='church'/>
<interp id='ref-serv'   value='servants'/>
<interp id='ref-cook'   value='cooking'/>
<!-- ... -->
</interpGrp>
</p></div1>
```

Once these interpretation elements have been defined, they can be linked with the parts of the text to which they apply in either or both of two ways. The ana attribute can be used on whichever element is appropriate:

```
<div1 type="chapter" n='38'>
<p id='P38.1' ana='set-church set-kitch'></p>
<s id="P38.1.1" ana='fig-apos'>Reader, I married him.</s>
...
```

Note in this example that since the paragraph has two settings (in the church and in the kitchen), the identifiers of both have been supplied.

Alternatively, the `<interp>` elements can point to all the parts of the text to which they apply, using their inst attribute:

```
<interp id='fig-apos' type='figure of speech' resp='LB, MSM'
   value='apostrophe' inst='P38.1.1'/>
<!-- ... -->
<interp id='set-church'  type='scene-setting' value='church'
   inst='P38.1' resp='LB, MSM'/>
<interp id='set-kitchen' type='scene-setting' value='kitchen'
   inst='P38.1' resp='LB, MSM'/>
<!-- ... -->
```

The `<interp>` is not limited to any particular type of analysis, The literary analysis shown above is but one possibility; one could equally well use `<interp>` to capture a linguistic part-of-speech analysis. For example, the example sentence given in section 8.3 (Linking Attributes) assumes a linguistic analysis which might be represented as follows:

```
<interp id="NP1" type="pos" value='noun phrase, singular'/>
<interp id="VV1" type="pos" value='inflected verb, present-tense singular'/>
...
```

17. Technical Documentation

Although the focus of this document is on the use of the TEI scheme for the encoding of existing ‘pre-electronic’ documents, the same scheme may also be used for the encoding of new documents. In the preparation of new documents (such as this one), SGML has much to recommend it: the document’s structure can be clearly represented, and the same electronic text can be re-used for many purposes --- to provide both online hypertext or browsable versions and well-formatted typeset versions from a common SGML source for example.

To facilitate this, a small number of additional elements are included in TEI Lite as extensions of the main TEI DTD, for use in marking particular features of technical documents in general, and of SGML-related documents in particular.

17.1. Additional Elements for Technical Documents

The following elements may be used to mark particular features of technical documents:

- `<eg>` contains a single short example of some technical topic being discussed, e.g. a code fragment or a sample of SGML encoding.
- `<code>` contains a short fragment of code in some formal language (often a programming language).
- `<ident>` contains an identifier of some kind, e.g. a variable name or the name of an SGML element or attribute.
- `<gi>` contains a special type of identifier: an SGML generic identifier, or element name.
- `<kw>` contains a keyword in some formal language.
- `<formula>` contains a mathematical or chemical formula, optionally presented in some non-SGML notation. Attributes include:
  - **notation** specifies the notation used to represent the body of the formula. Default value is *tex*, meaning the formula is represented using the TeX typesetting system.

The following example shows how these elements might be used to encode a passage from a tutorial introducing the Fortran programming language:

```
<p>It is traditional to introduce a language with a program like the
following:
<eg>
   CHAR*12 GRTG
   GRTG = ’HELLO WORLD’
   PRINT *, GRTG
   END
</eg></p>
<p>This simple example first declares a variable <ident>GRTG</ident>, in
the line <code>CHAR*12 GRTG</code>, which identifies <ident>GRTG</ident>
as consisting of 12 bytes of type <kw>CHAR</kw>.  To this variable,
the value <mentioned>HELLO WORLD</mentioned>
is then assigned. This is followed by a <kw>PRINT</kw> statement and an
<kw>END</kw> statement.
```

A formatting application, given a text like that above, can be instructed to format examples appropriately (e.g. to preserve line breaks, or to use a distinctive font). Similarly, the use of tags such as `<ident>` and `<kw>` greatly facilitates the construction of a useful index.

The `<formula>` element should be used to enclose a mathematical or chemical formula presented within the text as a distinct item. Since formulae generally include a large variety of special typographic features not otherwise present in ordinary text, it will usually be necessary to present the body of the formula in a specialized notation. The notation used should be specified by the notation attribute, as in the following example:

```
<formula notation="tex">
  \(E = mc^{2}\)
</formula>
```

The Tex notation is pre-defined for the TEI Lite DTD; other notations may be used if desired, but they must first be defined by a *notation* declaration within the DTD.

Almost any sequence of characters is permitted within the body of a `<formula>` element, as far as an SGML-aware processor is concerned. The data is passed unchanged by the parser to whatever application has been associated with the notation specified. The only exception to this rule is that the parser *will* recognize anything that resembles the start of an SGML end-tag, i.e. the character less-than (@) followed immediately by a solidus (/) and an alphabetic character. The following imaginary example would thus cause a confusing sequence of SGML parser errors:

```
<formula notation="tex">
  \(@/E = mc^{2}\)</formula>
```

Fortunately, the sequence @/ is quite unlikely to occur in most mathematical notations in practical use: if it does occur, special steps must be taken which are beyond the scope of this document (see the full Guidelines for more information).

This problem exists in a more acute form when SGML encoding is the subject of discussion within a technical document, itself encoded in SGML. In such a document, it is clearly essential to distinguish clearly the SGML markup occurring within examples from that marking up the document itself, and end-tags are highly likely to occur. The most general solution is to mark off the body of each SGML example as containing data which is not to be scanned for SGML mark-up by the parser. This is achieved by enclosing it within a special SGML construct called a *CDATA marked section*, as in the following example:

```
<p>A list should be encoded as follows:
<eg><![ CDATA [
   <list>
   <item>First item in the list</item>
   <item>Second item</item>
   </list>
]]>
</eg>
The <gi>list</gi> element consists of a series of <gi>item</gi>
elements.
```

The `<list>` element used within the example above will not be regarded as forming part of the document proper, because it is embedded within a marked section (beginning with the special markup declaration *@![CDATA [* , and ending with *]]>*).

Note also the use of the `<gi>` element to tag references to SGML element names (or *generic identifiers*) within the body of the text.

17.2. Generated Divisions

Most modern document production systems have the ability to generate automatically whole sections such as a table of contents or an index. The TEI Lite scheme provides an element to mark the location at which such a generated section should be placed.

`<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 `<divGen>` element can be placed anywhere that a division element would be legal, as in the following example:

```
<front>
<titlePage> ... </titlePage>
```

```
<divGen type="toc"/>
<div type='Preface'><head>Preface</head> ... </div>
</front>
<body> ... </body>
<back>
<div1><head>Appendix</head> ... </div1>
<divGen type="index" n='Index'/>
</back>
```

This example also demonstrates the use of the type attribute to distinguish the different kinds of division to be generated: in the first case a table of contents (a *toc*) and in the second an index.

When an existing index or table of contents is to be encoded (rather than one being generated) for some reason, the `<list>` element discussed in section 12 (Lists) should be used.

17.3. Index Generation

While production of a table of contents from a properly tagged document is generally unproblematic for an automatic processor, the production of a good quality index will often require more careful tagging. It may not be enough simply to produce a list of all parts tagged in some particular way, although extracting (for example) all occurrences of elements such as `<term>` or `<name>` will often be a good departure point for an index.

The TEI DTD provides a special purpose `<index>` tag which may be used to mark both the parts of the document which should be indexed, and how the indexing should be done.

`<index>` marks a location to be indexed for some purpose. Attributes include:

level1	gives the main form of the index entry.
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.

For example, the second paragraph of this section might include the following:

```
...
TEI lite also provides a special purpose <gi>index</gi> tag
<index level1='indexing'/>
<index level1='index (tag)' level2='use in index generation'/>
which may be used ...
```

The `<index>` element can also be used to provide a form of interpretive or analytic information. For example, in a study of Ovid, it might be desired to record all the poet's references to different figures, for comparative stylistic study. In the following lines of the *Metamorphoses*, such a study would record the poet's references to Jupiter (as *deus*, *se*, and as the subject of *confiteor* [in inflectional form number 227]), to Jupiter-in-the-guise-of-a-bull (as *imago tauri fallacis* and the subject of *teneo*), and so on.[1]

```
<l n="3.001">iamque deus posita fallacis imagine tauri
<l n="3.002">se confessus erat Dictaeaque rura tenebat</l>
```

This need might be met using the `<note>` element discussed in section in 7 (Notes), or with the `<interp>` element discussed in section 16 (Interpretation and Analysis). Here we demonstrate how it might also be satisfied by using the `<index>` element.

---

[1] The analysis is taken, with permission, from Willard McCarty and Burton Wright, *An Analytical Onomasticon to the Metamorphoses of Ovid* (Princeton: Princeton University Press, forthcoming). Some simplifications have been undertaken.

We assume that the object is to generate more than one index: one for names of deities (called dn), another for onomastic references (called on), a third for pronominal references (called pr) and so forth. One way of achieving this might be as follows:

```
<l n="3.001">iamque deus posita fallacis imagine tauri
     <index index="dn" level1="Iuppiter" level2="deus"/>
     <index index="on" level1="Iuppiter (taurus)"
                       level2="imago tauri fallacis"/></l>
<l n="3.002">se confessus erat Dictaeaque rura tenebat
     <index index="pr"    level1="Iuppiter" level2="se"/>
     <index index="v"     level1="Iuppiter" level2="confiteor (v227)"/>
     <index index="mons"  level1="Dicte" level2="rura Dictaea"/>
     <index index="regio" level1="Creta" level2="rura Dictaea"/>
     <index index="v"     level1="Iuppiter (taurus)"
                          level2="teneo (v9)"/></l>
```

For each `<index>` element above, an entry will be generated in the appropriate index, using as headword the value of the level1 attribute, and as secondary keyword that of the level2 attribute, which contains the word cited in nominative form. The actual reference will be taken from the context in which the `<index>` element appears, i.e. in this case the identifier of the `<l>` element containing it.

18. Character Sets, Diacritics, etc.

For those working with standard forms of the European languages, the TEI recommendations for character set use are simple. For local use, use whatever character set is supported by your machine and your software. If your software makes direct keyboard entry of special characters difficult, you may elect to define your own keyboarding conventions (for example to represent accented letters by typing the appropriate accent immediately after the letter, or by using special sequences unlikely to appear in normal text such as *aE* for *ä*). Global search and replace functions can then be used to turn these keyboard shorthands into the proper characters. If you work with non-Latin scripts and there is a standard transliteration scheme in your field (e.g. for ancient Greek the beta code of the Thesaurus Linguæ Græcæ), use it. Any transliteration used should be reversible (this rules out a surprising number of schemes commonly used in normal writing) and will be most usable if it requires no special ligatures, ties, or diacritics (this rules out a surprising number of the remainder).

For interchange of files among systems, use SGML entity references to replace all characters not in the following list of characters which almost always survive electronic interchange intact:

```
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
" % & ' ( ) * + , - . / : ; < = > ? _   (space)
```

This list excludes the following characters which, to the frequent annoyance of unsuspecting users, often do not survive transfer across national boundaries or over standard wide-area networks. If you're just going from your Mac to your PC, though, these characters will probably be safe:

```
! # $ [ \ ] ^ ` { } | ~
```

To ensure proper transmission across multi-vendor networks, entity references must be used for all accented and extended-Latin characters, all non-Latin characters, and all symbols not on conventional computer keyboards.

You may use your own SGML entity names in TEI-conformant files, if you wish and if you provide standard SGML entity declarations for them, but the standard names (though long-winded) have the advantage of clarity; the characters intended are reasonably clear to any speaker of English who recognizes that a character is being named, often even without recourse to any list. This is not true of many other schemes for representing accented characters.

The entity names needed for the characters listed above as ‘unsafe’ and for the accented characters of some major Western European languages are given below. Lists of public entity sets and their contents are available in any reference work on SGML: the names given below are from ISO public entity sets, are widely used, and are therefore recommended.

When the character you need does not appear in the public entity sets, you may wish to generate a name using the same naming conventions used in ISO public entity sets, as described here:

digraphs Form entity names for digraphs by appending the string *lig* to the letters forming the digraph. If a capitalized form is required, both letters are given in upper case (remember that case is usually significant in entity names). E.g.: *aelig* (æ), *AElig* (Æ) *szlig* (ß).

diacritics and accents Form entity names for accented letters in most Western European languages by appending one of the following strings to the letter bearing the accent, which may be in upper or lower case.

umlaut use *uml* for umlaut or trema: e.g. *auml* (ä), *Auml* (Ä), *euml* (ë), *iuml* (sic: ï), *ouml* (ö), *Ouml* (Ö), *uuml* (ü), *Uuml* (Ü).

acute use *acute* for acute or stressed accent: e.g. *aacute* (á), *eacute* (é), *Eacute* (É), *iacute* (í), *oacute* (ó), *uacute* (ú).

grave use *grave* for grave accent: e.g. *agrave* (à), *egrave* (è), *igrave* (ì), *ograve* (ò), *ugrave* (ù).

circumflex use *circ* for circumflex: e.g. *acirc* (â), *ecirc* (ê), *Ecirc* (Ê), *icirc* (î), *ocirc* (ô), *ucirc* (û).

tilde use *tilde* for tilde: e.g. *atilde* (ã), *Atilde* (Ã), *ntilde* (ñ), *Ntilde* (Ñ), *otilde* (õ), *Otilde* (Õ).

consonants The following are recommended entity names for some special consonants found in Western European languages: *ccedil* (ç), *Ccedil* (Ç), *eth* (lowercase eth or Anglo-Saxon/Icelandic crossed d), *ETH* (uppercase eth), *thorn* (lowercase thorn), *THORN* (uppercase thorn), *szlig* (German s-z ligature or *esszett*, ß).

punctuation marks The following are recommended entity names for some commonly found punctuation marks: *ldquo* (left double quotation mark, in shape of superscript 66), *rdquo* (right double quotation mark, superscript 99), *mdash* (one-em dash), *hellip* (horizontal ellipsis, three closely spaced dots), *rsquo* (right single quote, in shape of superscript 9). See also the list of ‘unsafe’ characters given below.

‘unsafe’ characters The characters listed above as unsafe for transmission over current international academic and public-access networks may be represented with the following entities: *excl* (!), *num* (#), *dollar* ($), *lsqb* (left square bracket), *bsol* (back-slanted solidus, \), *rsqb* (right square bracket), *circ* (circumflex, ^), *lsquo* (left single quotation mark), *grave* (grave accent), *lcub* (left curly bracket, {), *rcub* (right curly bracket, }), *verbar* (vertical bar, |), *tilde* (~).

19. Front and Back Matter

19.1. Front Matter

For many purposes, particularly in older texts, the preliminary material such as title pages, prefatory epistles, etc., may provide very useful additional linguistic or social information. P3 provides a set of recommendations for distinguishing the textual elements most commonly encountered in front matter, which are summarized here.

19.1.1. Title Page

The start of a title page should be marked with the element `<titlePage>`. All text contained on the page should be transcribed and tagged with the appropriate element from the following list:

`<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. Must be divided into `<titlePart>` elements.
`<titlePart>`	contains a subsection or division of the title of a work, as indicated on a title page; also used for free-floating fragments of the title page not part of the document title, authorship attribution, etc. Attributes include: **type** specifies the role of this subdivision of the title. Suggested values include: *main* (main title), *sub* (subtitle), *desc* (a descriptive paraphrase of the work included in the title), and *alt* (alternative 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>`).
`<docDate>`	contains the date of the document, as given (usually) on the title page.
`<docEdition>`	contains an edition statement as presented on a title page of a document.
`<docImprint>`	contains the imprint statement (place and date of publication, publisher name), as given (usually) at the foot of a title page.
`<epigraph>`	contains a quotation, anonymous or attributed, appearing at the start of a section or chapter, or on a title page.

Typeface distinctions should be marked with the **rend** attribute when necessary, as described above. Very detailed description of the letter spacing and sizing used in ornamental titles is not as yet provided for by the Guidelines. Changes of language should be marked by appropriate use of the **lang** attribute or the `<foreign>` element, as necessary. Names, wherever they appear, should be tagged using the `<name>`, as elsewhere.

Two example title pages follow:

```
<titlePage rend="Roman">
  <docTitle><titlePart type="main">
    PARADISE REGAIN’D. A POEM In IV <hi>BOOKS</hi>.
    </titlePart>
    <titlePart>
    To which is added <title>SAMSON AGONISTES</title>.
    </titlePart>
  </docTitle>
  <byLine>The Author <docAuthor>JOHN MILTON</docAuthor></byline>
  <docImprint><name>LONDON</name>,
    Printed by <name>J.M.</name>
    for <name>John Starkey</name>
    at the <name>Mitre</name>
    in <name>Fleetstreet</name>,
    near <name>Temple-Bar.</name>
  </docImprint>
  <docDate>MDCLXXI</docDate>
</titlePage>
```

```
<titlePage>
  <docTitle><titlePart type="main">
  Lives of the Queens of England, from the Norman
    Conquest;</titlePart>
  <titlePart type=’sub’>with anecdotes of their courts.
  </titlePart></docTitle>
  <titlePart>Now first published from Official Records
    and other authentic documents private as well as
    public.</titlePart>
  <docEdition>New edition, with corrections and
    additions</docEdition>
  <byline>By <docAuthor>Agnes Strickland</docAuthor></byline>
  <epigraph>
    <q>The treasures of antiquity laid up in old
       historic rolls, I opened.</q>
    <bibl>BEAUMONT</bibl>
  </epigraph>
  <docImprint>Philadelphia: Blanchard and Lea</docImprint>
  <docDate>1860.</docDate>
</titlePage>
```

19.1.2. Prefatory Matter

Major blocks of text within the front matter should be marked as `<div>` or `<div1>` elements; the following suggested values for the type attribute may be used to distinguish various common types of prefatory matter:

foreword a text addressed to the reader, by the author, editor or publisher, possibly in the form of a letter.

preface a text addressed to the reader, by the author, editor or publisher, possibly in the form of a letter.

dedication a text (often a letter) addressed to someone other than the reader in which the author typically commends the work in hand to the attention of the person concerned.

abstract a prose argument summarizing the content of the work.

ack Acknowledgements.

contents a table of contents (typically this should be tagged as a `<list>`).

frontispiece a pictorial frontispiece, possibly including some text.

Like any text division, those in front matter may contain low level structural or non-structural elements as described elsewhere. They will generally begin with a heading or title of some kind which should be tagged using the `<head>` element. Epistles will contain the following additional elements:

`<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.

`<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.

`<argument>` A formal list or prose description of the topics addressed by a subdivision of a text.

`<cit>` A quotation from some other document, together with a bibliographic reference to its source.

`<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.

Epistles which appear elsewhere in a text will, of course, contain these same elements.

As an example, the dedication at the start of Milton’s *Comus* should be marked up as follows:

```
<div type=’dedication’>
<head>To the Right Honourable <name>JOHN Lord Viscount
BRACLY</name>, Son and Heir apparent to the Earl of
Bridgewater, &amp;c.</head>
<salute>MY LORD,</salute>

<p>THis <hi>Poem</hi>, which receiv’d its first occasion of
Birth from your Self, and others of your Noble Family ....
and as in this representation your attendant
<name>Thyrsis</name>, so now in all reall expression
<closer>
<salute>Your faithfull, and most humble servant</salute>
<signed><name>H. LAWES.</name></signed>
</closer>
</div>
```

19.2. Back Matter

19.2.1. Structural Divisions of Back Matter

Because of variations in publishing practice, back matter can contain virtually any of the elements listed above for front matter, and the same elements should be used where this is so. Additionally, back matter may contain the following types of matter within the `<back>`

element. Like the structural divisions of the body, these should be marked as `<div>` or `<div1>` elements, and distinguished by the following suggested values of the type attribute:

appendix an appendix.

glossary a list of words and definitions, typically in the form of a `list type=gloss`.

notes a series of `<note>`s.

bibliography a series of bibliographic references, typically in the form of a special bibliographic-list element `<listBibl>`, whose items are individual `<bibl>` elements.

index a set of index entries, possibly represented as a structured list or glossary list, with optional leading `<head>` and perhaps some paragraphs of introductory or closing text (TEI P3 defines other specialized elements for generating indices in document production, described above in section 17.3 (Index Generation)).

colophon a description at the back of the book describing where, when, and by whom it was printed; in modern books it also often gives production details and identifies the type faces used.

20. The Electronic Title Page

Every TEI text has a header which provides information analogous to that provided by the title page of printed text. The header is introduced by the element `<teiHeader>` and has four major parts:

`<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.

A corpus or collection of texts, which share many characteristics, may have one header for the corpus and individual headers for each component of the corpus. In this case the type attribute indicates the type of header.

```
<teiHeader type="corpus">
```

introduces the header for corpus-level information.

Some of the header elements contain running prose which consists of one or more `<p>`s. Others are grouped:

- Elements whose names end in *Stmt*(for statement) usually enclose a group of elements recording some structured information.
- Elements whose names end in *Decl* (for declaration) enclose information about specific encoding practices.
- Elements whose names end in *Desc* (for description) contain a prose description.

20.1. The File Description

The `<fileDesc>` element is mandatory. It contains a full bibliographic description of the file with the following elements:

`<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 minimal header has the following structure:

```
<teiHeader>
     <fileDesc>
          <titleStmt> ... </titleStmt>
          <publicationStmt> ... <publicationStmt>
          <sourceDesc> ... <sourceDesc>
     </fileDesc>
</teiHeader>
```

20.1.1. The Title Statement

The following elements can be used in the `<titleStmt>`:

`<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.

It is recommended that the title should distinguish the computer file from the source text, for example:

```
[title of source]: a machine readable transcription
[title of source]: electronic edition
A machine readable version of: [title of source]
```

The `<respStmt>` element contains the following subcomponents:

- `<resp>` contains a phrase describing the nature of a person's intellectual responsibility.
- `<name>` contains a proper noun or noun phrase.

Example:

```
<titleStmt>
     <title>Two stories by Edgar Allen Poe: a machine readable
               transcription</title>
     <author>Poe, Edgar Allen (1809-1849)
     <respStmt><resp>compiled by</resp>
     <name>James D. Benson</name></respStmt>
</titleStmt>
```

20.1.2. The Edition Statement

The `<editionStmt>` groups information relating to one edition of a text (where *edition* is used as elsewhere in bibliography), and may include the following elements:

- `<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.

Example:

```
<editionStmt>
     <edition n="U2">Third draft, substantially revised
     <date>1987</date>
     </edition>
</editionStmt>
```

Determining exactly what constitutes a new edition of an electronic text is left to the encoder.

20.1.3. The Extent Statement

The `<extent>` statement describe the approximate size of a file.

Example:

```
<extent>4532 bytes</extent>
```

20.1.4. The Publication Statement

The `<publicationStmt>` is mandatory. It may contain 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.

At least one of these three elements must be present, unless the entire publication statement is in prose. The following elements may occur within them:

- `<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 `restricted`, `unknown`, and `free`.

`<date>` contains a date in any format.

Example:

```
<publicationStmt>
     <publisher>Oxford University Press</publisher>
     <pubPlace>Oxford</pubPlace> <date>1989</date>
     <idno type="ISBN"> 0-19-254705-5</idno>
     <availability>Copyright 1989, Oxford University
          Press</availability>
</publicationStmt>
```

20.1.5. Series and Notes Statements

The `<seriesStmt>` groups information about the series, if any, to which a publication belongs. It may contain `<title>`, `<idno>`, or `<respStmt>` elements.

The `<notesStmt>`, if used, contains one or more `<note>` elements which contain a note or annotation. Some information found in the notes area in conventional bibliography has been assigned specific elements in the TEI scheme.

20.1.6. The Source Description

The `<sourceDesc>` is a mandatory element which records details of the source or sources from which the computer file is derived. It may contain simple prose or a bibliographic citation, using one or more of the following elements:

`<bibl>` contains a loosely-structured bibliographic citation of which the subcomponents 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.

`<listBibl>` contains a list of bibliographic citations of any kind.

Examples:

```
<sourceDesc>
     <bibl>The first folio of Shakespeare, prepared by Charlton
          Hinman (The Norton Facsimile, 1968)</bibl>
</sourceDesc>
```

```
<sourceDesc>
     <scriptStmt id="CNN12">
     <bibl><author>CNN Network News
          <title>News headlines
          <date>12 Jun 1989
```

```
        </bibl>
        </scriptStmt>
   </sourceDesc>
```

20.2. The Encoding Description

The `<encodingDesc>` element specifies the methods and editorial principles which governed the transcription of the text. Its use is highly recommended. It may be prose description or may contain elements from the following list:

`<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.

20.2.1. Project and Sampling Descriptions

Examples of `<projectDesc>` and `<samplingDesc>`:

```
   <encodingDesc>
        <projectDesc>Texts collected for use in the Claremont
             Shakespeare Clinic, June 1990.
        </projectDesc>
   </encodingDesc>
```

```
   <encodingDesc>
        <samplingDecl>Samples of 2000 words taken from the beginning
             of the text
        </samplingDecl>
   </encodingDesc>
```

20.2.2. Editorial Declarations

The `<editorialDecl>` contains a prose description of the practices used when encoding the text. Typically this description should cover such topics as the following, each of which may conveniently be given as a separate paragraph.

correction	how and under what circumstances corrections have been made in the text.
normalization	the extent to which the original source has been regularized or normalized.
quotation	what has been done with quotation marks in the original -- have they been retained or replaced by entity references, are opening and closing quotes distinguished, etc.
hyphenation	what has been done with hyphens (especially end-of-line hyphens) in the original -- have they been retained, replaced by entity references, etc.

segmentation how has the text has been segmented, for example into sentences, tone-units, graphemic strata, etc.

interpretation what analytic or interpretive information has been added to the text.

Example:

```
<editorialDecl>
          <p>The part of speech analysis applied throughout
               section 4 was added by hand and has not been
               checked.
          <p>Errors in transcription controlled by using the
               WordPerfect spelling checker.
          <p>All words converted to Modern American spelling
               using Webster's 9th Collegiate dictionary.
          <p>All quotation marks converted to entity
               references &odq; and &cdq;.
</editorialDecl>
```

20.2.3. Tagging, Reference, and Classification Declarations

The `<tagsDecl>` element is used to provide detailed information about the SGML tags actually appearing within a text. It may contain a simple list of elements used, with a count for each, using the following special purpose elements:

`<tagUsage>` supplies information about the usage of a specific element within the outermost `<text>` of a TEI conformant document. Attributes include:

- **gi** the name (generic identifier) of the element indicated by the tag.
- **occurs** specifies the number of occurrences of this element within the text.

The `<rendition>` element is used to document different ways in which elements are rendered in the source text.

`<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.

For example:

```
<tagsDecl>
 <tagUsage gi="text" occurs=1>
 <tagUsage gi="body" occurs=1>
 <tagUsage gi=p occurs="12">
 <tagUsage gi="hi" occurs=6>
</tagsDecl>
```

This (imaginary) tags declaration would be appropriate for a text containing twelve paragraphs in its body, within which six `<hi>` elements have been marked. Note that if the `<tagsDecl>` element is used, it must contain a `<tagUsage>` element for *every* element tagged in the associated text element.

The `<refsDecl>` element is used to document the way in which any standard referencing scheme built into the encoding works. In its simplest form, it consists of prose description.

Example:

```
<refsDecl>
     <p>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 numeral and
     yyy is the section number in arabic.
</refsDecl>
```

The `<classDecl>` element groups together definitions or sources for any descriptive classification schemes used by other parts of the header. At least one such scheme must be provided, encoded using the following elements:

`<taxonomy>`	defines a typology used to classify texts either implicitly, by means of a bibliographic citation, or explicitly by a structured taxonomy.
`<bibl>`	contains a loosely-structured bibliographic citation of which the subcomponents may or may not be explicitly tagged.
`<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, in the form of a brief prose description.

In the simplest case, the taxonomy may be defined by a bibliographic reference, as in the following example:

```
<classDecl>
     <taxonomy id='LCSH'>
          <bibl>Library of Congress Subject Headings
          </bibl>
     </taxonomy>
</classDecl>
```

Alternatively, or in addition, the encoder may define a special purpose classification scheme, as in the following 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>
```

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 further described below.

20.3. *The Profile Description*

The `<profileDesc>` element enables information characterizing various descriptive aspects of a text to be recorded within a single framework. It has three optional components:

- `<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.

Examples:

```
<creation>
     <date value='1992-08'>August 1992</date>
     <name type="place">Taos, New Mexico</name>
</creation>
```

The `<textClass>` element classifies a text by reference to the system or systems defined by the `<classDecl>` element, and contains one or more of the following elements:

- `<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 element `<keywords>` contains a list of keywords or phrases identifying the topic or nature of a text. The attribute scheme links these to the classification system defined in `<taxonomy>`.

```
<textClass>
     <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>
          </list>
     </keywords>
</textClass>
```

20.4. *The Revision Description*

The `<revisionDesc>` element provides a change log in which each change made to a text may be recorded. The log may be recorded as a sequence of `<change>` elements each of which contains

- `<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.

Example:

```
<revisionDesc>
     <change><date>6/3/91:</date>
          <respStmt><name>EMB</name><resp>ed.</resp></respStmt>
          <item>File format updated</item></change>
     <change><date>5/25/90:</date>
          <respSmt><name>EMB</name><resp>ed.</resp>
          <item>Stuart's corrections entered</item></change>
</revisionDesc>
```

Appendix A List of Elements Described

Appendix A.1 Global Attributes

All elements in the TEI Lite document type definition have the following global attributes:

ana	links an element with its interpretation.
corresp	links an element with one or more other corresponding elements.
id	Unique identifier for the element; must begin with a letter, can contain letters, digits, hyphens, and periods.
lang	language of the text in this element; if not specified, language is assumed to be the same as in the surrounding context.
n	Name or number of this element; may be any string of characters. Often used for recording traditional reference systems.
next	links an element to the next element in an aggregate.
prev	links an element to the previous element in an aggregate.
rend	physical realization of the element in the copy text: `italic`, `roman`,`display block`, etc. Value may be any string of characters.

Appendix A.2 Elements in TEI Lite

The following list shows all the elements defined for the TEI Lite DTD, with a brief description of each:

`<abbr>`	contains an abbreviation of any sort; expansion may be given in the **expan** attribute.
`<add>`	contains letters, words, or phrases inserted in the text by an author, scribe, annotator, or corrector.
`<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.
`<anchor>`	specifies a location or point within a document so that it may be pointed to.
`<argument>`	A formal list or prose description of the topics addressed by a subdivision of a text.
`<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.
`<authority>`	supplies the name of a person or other agency responsible for making an electronic file available, other than a publisher or distributor.
`<availability>`	supplies information about the availability of a text, for example any restrictions on its use or distribution, its copyright status, etc.
`<back>`	contains any appendixes, etc., following the main part of a text.
`<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.

`<biblScope>` defines the scope of a bibliographic reference, for example as a list of page numbers, or a named subdivision of a larger work.

`<body>` contains the whole body of a single unitary text, excluding any front or back matter.

`<byline>` contains the primary statement of responsibility given for a work on its title page or at the head or end of the work.

`<catDesc>` describes some category within a taxonomy or text typology, in the form of a brief prose description.

`<category>` contains an individual descriptive category, possibly nested within a superordinate category, within a user-defined taxonomy.

`<catRef>` specifies one or more defined categories within some taxonomy or text typology.

`<cell>` contains one cell of a table.

`<cit>` A quotation from some other document, together with a bibliographic reference to its source.

`<classCode>` contains the classification code used for this text in some standard classification system, which is identified by the scheme attribute.

`<classDecl>` contains one or more taxonomies defining any classificatory codes used elsewhere in the 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.

`<code>` contains a short fragment of code in some formal language (often a programming language).

`<corr>` contains the correct form of a passage apparently erroneous in the copy text.

`<creation>` contains information about the creation of a text.

`<date>` contains a date in any format, with normalized value in the value attribute.

`<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.

`<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.

`<distributor>` supplies the name of a person or other agency responsible for the distribution of a text.

`<div>` contains a subdivision of the front, body, or back of a text.

`<div1>` **...** `<div7>` contains a first-, second, ..., seventh-level subdivision of the front, body, or back of a text.

`<divGen>` indicates the location at which a textual division generated automatically by a text-processing application is to appear; the type attribute specifies whether it is an index, table of contents, or something else.

`<docAuthor>` contains the name of the author of the document, as given on the title page (often but not always contained in a `<byline>`).

`<docDate>` contains the date of the document, as given (usually) on the title page.

`<docEdition>` contains an edition statement as presented on a title page of a document.

`<docImprint>` contains the imprint statement (place and date of publication, publisher name), as given (usually) at the foot of a title page.

`<docTitle>` contains the title of a document, including all its constituents, as given on a title page. Must be divided into `<titlePart>` elements.

`<edition>` describes the particularities of one edition of a text.

`<editionStmt>` groups information relating to one edition of a text.

`<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.

`<editorialDecl>` provides details of editorial principles and practices applied during the encoding of a text.

`<eg>` contains a single short example of some technical topic being discussed, e.g. a code fragment or a sample of SGML encoding.

`<emph>` marks words or phrases which are stressed or emphasized for linguistic or rhetorical effect.

`<encodingDesc>` documents the relationship between an electronic text and the source or sources from which it was derived.

`<epigraph>` contains a quotation, anonymous or attributed, appearing at the start of a section or chapter, or on a title page.

`<extent>` describes the approximate size of the electronic text as stored on some carrier medium, specified in any convenient units.

`<figure>` marks the spot at which a graphic is to be inserted in a document. Attributes may be used to indicate an SGML entity containing the image itself (in some non-SGML notation); paragraphs within the `<figure>` element may be used to transcribe captions.

`<fileDesc>` contains a full bibliographic description of an electronic file.

`<foreign>` identifies a word or phrase as belonging to some language other than that of the surrounding text.

`<formula>` contains a mathematical or chemical formula, optionally presented in some non-SGML notation. The notation is used to name the non-SGML notation used to transcribe the formula.

`<front>` contains any prefatory matter (headers, title page, prefaces, dedications, etc.) found before the start of a text proper.

`<funder>` specifies the name of an individual, institution, or organization responsible for the funding of a project or text.

`<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.

`<gi>` contains a special type of identifier: an SGML generic identifier, or element name.

`<gloss>` marks a word or phrase which provides a gloss or definition for some other word or phrase.

`<group>` contains a number of unitary texts or groups of texts.

`<head>` contains any heading, for example, the title of a section, or the heading of a list or glossary.

`<hi>` marks a word or phrase as graphically distinct from the surrounding text, for reasons concerning which no claim is made.

`<ident>` contains an identifier of some kind, e.g. a variable name or the name of an SGML element or attribute.

`<idno>` supplies any standard or non-standard number used to identify a bibliographic item; the type attribute identifies the scheme or standard.

`<imprint>` groups information relating to the publication or distribution of a bibliographic item.

`<index>` marks a location to be indexed for some purpose. Attributes are used to give the main form, and second- through fourth-level forms to be entered in the index indicated.

`<interp>` provides for an interpretive annotation which can be linked to a span of text. Attributes include resp, type, and value.

`<interpGrp>` collects together `<interp>` tags.

`<item>` contains one component of a list.

`<keywords>` contains a list of keywords or phrases identifying the topic or nature of a text; if the keywords come from a controlled vocabulary, it can be identified by the scheme attribute.

`<kw>` contains a keyword in some formal language.

`<l>` contains a single, possibly incomplete, line of verse.

`<label>` contains the label associated with an item in a list; in glossaries, marks the term being defined.

`<langUsage>` describes the languages, sublanguages, registers, dialects, etc., represented within a text.

`<lb>` marks the start of a new (typographic) line in some edition or version of a text.

`<lg>` contains a group of verse lines functioning as a formal unit e.g. a stanza, refrain, verse paragraph, etc.

`<list>` contains any sequence of items organized as a list, whether of numbered, bulletted, or other type.

`<listBibl>` contains a list of bibliographic citations of any kind.

`<mentioned>` marks words or phrases mentioned, not used.

`<milestone>` marks the boundary between sections of a text, as indicated by changes in a standard reference system. Attributes include ed (edition), unit (page, etc.), and n (new value).

`<name>` contains a proper noun or noun phrase. Attributes can indicate its type, give a normalized form, or associate it with a specific individual or thing by means of a unique identifiers.

`<note>` contains a note or annotation, with attributes to indicate the type, location, and source of the note.

`<notesStmt>` collects together any notes providing information about a text additional to that recorded in other parts of the bibliographic description.

`<num>` contains a number, written in any form, with normalized value in the value attribute.

`<opener>` groups together dateline, byline, salutation, and similar phrases appearing as a preliminary group at the start of a division, especially of a letter.

`<orig>` contains the original form of a reading, for which a regularized form may be given in the attribute reg.

`<p>` marks paragraphs in prose.

`<pb>` marks the boundary between one page of a text and the next in a standard reference system.

`<principal>` supplies the name of the principal researcher responsible for the creation of an electronic text.

`<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.

`<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.

`<ptr>` a pointer to another location in the current document in terms of one or more identifiable elements.

`<publicationStmt>` groups information concerning the publication or distribution of an electronic or other text.

`<publisher>` provides the name of the organization responsible for the publication or distribution of a bibliographic item.

`<pubPlace>` contains the name of the place where a bibliographic item was published.

`<q>` contains a quotation or apparent quotation.

`<ref>` a reference to another location in the current document, in terms of one or more identifiable elements, possibly modified by additional text or comment.

`<refsDecl>` specifies how canonical references are constructed for this text.

`<reg>` contains a reading which has been regularized or normalized in some sense; original reading may be given in the attribute orig.

`<rendition>` supplies information about the intended rendition of one or more elements.

`<resp>` contains a phrase describing the nature of a person's intellectual responsibility.

`<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.

`<revisionDesc>` summarizes the revision history for a file.

`<row>` contains one row of a table.

`<rs>` contains a general purpose name or referring string. Attributes can indicate its type, give a normalized form, or associate it with a specific individual or thing by means of a unique identifiers.

`<s>` identifies an *s-unit* within a document, for purposes of establishing a simple canonical referencing scheme covering the entire text.

`<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.

`<samplingDecl>` contains a

prose description of the rationale and methods used in sampling texts in the creation of a corpus or collection.

`<seg>` identifies a span or segment of text within a document so that it may be pointed to; the type attribute categorizes the segment.

`<series>` contains information about the series in which a book or other bibliographic item has appeared.

`<seriesStmt>` groups information about the *series*, if any, to which a publication belongs.

`<sic>` contains text reproduced although apparently incorrect or inaccurate.

`<signed>` contains the closing salutation, etc., appended to a foreword, dedicatory epistle, or other division of a text.

`<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.

`<sourceDesc>` supplies a bibliographic description of the copy text(s) from which an electronic text was derived or generated.

`<sp>` contains an individual speech in a performance text, or a passage presented as such in a prose or verse text, with who attribute to identify speaker.

`<speaker>` contains a special form of heading or label, giving the name of one or more speakers in a performance text or fragment.

`<sponsor>` specifies the name of a sponsoring organization or institution.

`<stage>` contains any kind of stage direction within a performance text or fragment.

`<table>` contains text displayed in tabular form, in rows and columns.

`<tagsDecl>` provides detailed information about the tagging applied to an SGML document.

`<tagUsage>` supplies information about the usage of a specific element within the outermost `<text>` of a TEI conformant document.

`<taxonomy>` defines a typology used to classify texts either implicitly, by means of a bibliographic citation, or explicitly by a structured taxonomy.

`<term>` contains a single-word, multi-word or symbolic designation which is regarded as a technical term.

`<textClass>` groups information which describes the nature or topic of a text in terms of a standard classification scheme, thesaurus, etc.

`<time>` contains a phrase defining a time of day in any format, with normalized value in the value attribute.

`<title>` contains the title of a work, whether article, book, journal, or series, including any alternative titles or subtitles.

`<titlePage>` contains the title page of a text, appearing within the front or back matter.

`<titlePart>` contains a subsection or division of the title of a work, as indicated on a title page; also used for free-floating fragments of the title page not part of the document title, authorship attribution, etc.

`<titleStmt>` groups information about the title of a work and those responsible for its intellectual content.

`<trailer>` contains a closing title or footer appearing at the end of a division of a text.

`<unclear>` contains a word, phrase, or passage which cannot be transcribed with certainty because it is illegible or inaudible in the source.

`<xptr>` defines a pointer to another location in the current document or an external document.

`<xref>` defines a pointer to another location in the current document or an external document, possibly modified by additional text or comment.

Appendix B References

This appendix contains a list of bibliographic references for works on SGML and related topics, presented also to demonstrate the use of the `<bibl>` element discussed in section 13 (Bibliographic Citations) above.

```
<listBibl>

<bibl>ALA (American Library Association).  <title>ALA-
LC
Romanization Tables:  Transliteration Schemes for Non-
Roman
Scripts</title>, approved by the Library of Congress and the American

Library Association, tables compiled and edited by Randall K. Barry.

Washington:  Library of Congress, 1991.
</bibl>

<bibl>ANSI (American National Standards Institute).  <title>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).</title>  [New York]:  ANSI, 1986.

</bibl>

<bibl>
<author>Barnard, David, et al.</author>
<title level="a">SGML-
Based Markup for Literary Texts.</title>

<title>Computers and the Humanities</title>

<biblScope>22 (1988):  265-
76.</biblScope>
</bibl>

<bibl>
   <author>Barron, David</author>
   <title level="a">Why use SGML?</title>

   <title>Electronic Publishing
          Origination, Dissemination and Design</title>

   <biblScope>2.1 (April 1989):  3-
24.</biblScope>
</bibl>

<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>Communications of the

ACM</title>
<biblScope>30.11 (November 1987):  933-
947.</biblScope>
</bibl>

<bibl>
   <editor>Cover, Robin C., et al.</editor>
   <title>A Bibliography on Structured Text:
          Technical Report 90-
281</title>
   <publisher>Queen’s University,</publisher>
   <pubPlace>Kingston, Ont.</pubPlace>
   <date>June 1990</date>
<note place="inline">A current version of this bibliography
is maintained at <code>http://www.sil.org/sgml/sgml.html</code>.
</bibl>

<bibl>Goldfarb, Charles F., <title>The SGML Handbook</title>
Oxford:  Clarendon Press, 1990.</bibl>

<bibl>
   <author>van Herwijnen, Eric.</author>
   <title>Practical SGML.</title>
   <publisher>Kluwer Academic Publishers</publisher>
   <date>1990; 2d ed. 1994.</date>
</bibl>

<bibl>ISO (International Organization for Standardization).
<title>ISO 8859-
1:  1987 (E).  Information processing -
-- 8-bit
Single-
Byte Coded Graphic Character Sets ---
Part 1:  Latin Alphabet No.
1.</title>  (<title>Traitement de l’information
-- Jeux de caracte■res
graphiques cod&eacute;s sur un seul octet -
-- Partie 1:  Alphabet latin no
1.</title>) First edition --- 1987-02-
15.  [Geneva]:  International
Organization for Standardization, 1987.
</bibl>

<bibl>ISO (International Organization for Standardization).
<title>ISO 8879-
1986 (E).  Information processing ---
Text and Office
Systems ---
Standard Generalized Markup Language (SGML).</title>  First
edition --- 1986-10-
15.  [Geneva]:  International Organization for
Standardization, 1986.
</bibl>

<bibl>ISO (International Organization for Standardization).
<title>ISO 8879:1986 / A1:1988 (E).  Information processing -
-- Text and
Office Systems ---
Standard Generalized Markup Language (SGML),
Amendment 1.</title>  Published 1988-
07-01.
[Geneva]:  International Organization for Standardization, 1988.
</bibl>
<bibl>ISO (International Organization for Standardization).
<title>ISO/TR 9573-
1988(E).  Information processing---
SGML support
facilities ---
Techniques for using SGML.</title>  Final text of
1988-09-12.
</bibl>

<bibl>ISO (International Organization for Standardization), and IEC
(International Electrotechnical Commission). <title>ISO/IEC 10646-
1:
1993.  Information technology ---
Universal Multiple-Octet Coded
Character Set (UCS) ---
Part 1:  Architecture and Basic Multilingual
Plane.</title>
[Geneva]:  International Organization for
Standardization, 1993.
</bibl>
<bibl>ISO (International Organization for Standardization), and IEC
(International Electrotechnical Commission).
<title>ISO/IEC 10744: 1992.  Information
Technology --- Hypermedia/Time-
based Structuring Language
(HyTime).</title>
[Geneva]:  International Organization for Standardization, 1992.
</bibl>

<bibl>
Langendoen, D. Terence, and Gary F. Simons.
<title level="a">A Rationale for the TEI
Recommendations for Feature-
Structure Markup.</title>
<title>Computers and the Humanities</title>
(1995; in press).
</bibl>
<bibl>
```

```
<author>Warmer, J., and S. van Egmond</author>
<title level="a">The implementation of the Amsterdam
     SGML parser.</title>
<title>Electronic Publishing
     Origination, Dissemination and Design</title>
```

```
   <biblScope>2.2 (July 1989):  65-
90</biblScope>
</bibl>
</listBibl>
```