TEI and XML: exercises and resources for week 2

Sebastian Rahtz

Contents

1 Introduction

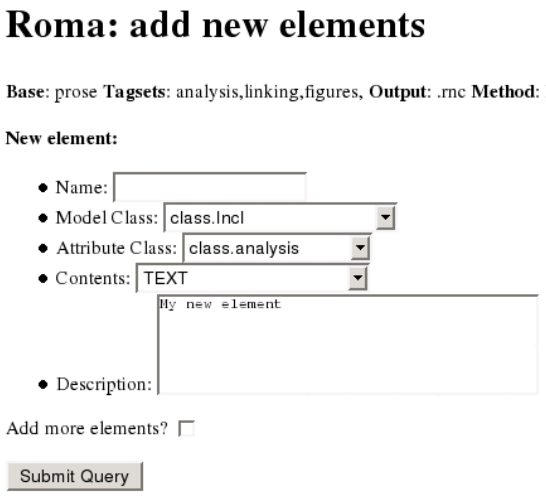
The example files used in these exercises are all available on the web at `http://www.tei-c.org.uk/Talks/OUCS/2004-02/samples2.zip`. Before starting work, you should download this file to your local disk and unpack it. Work on the H: drive on your PC.

In this exercise, you will use a tool on the TEI web site to make a XML schema tailored to you. You will need access to the net and a browser such as IE, Mozilla, or Opera. Once you have your schema you will also need an XML-aware editor to use it (in our case, Gnu Emacs).

Our goal is to make a very very simple schema, which we can use to mark up a multimedia document. We don't need anything like the full complexity of TEI Lite, much less the full TEI. We just want to mark up headings, dates, lists, paragraphs, figures and ... sound clips. Unfortunately, the TEI Guidelines don't seem to have an element specifically for marking up sound clips, so we need to invent it. We'll also have to say where it is allowed to appear.

2 Making your own schema

1. Go to the URL http://www.tei-c.org.uk/Roma/.
2. Choose the modules you want: for this exercise, select the prose base, the figures topping, and the linking topping. (This means you should uncheck one of the three toppings that the system offers you: analysis) If you want to read about a module in detail, you can click on its name to browse the full text of the relevant part of the TEI Guidelines.
3. The modules chosen contain many more elements than we need, so we will set things up to ignore most of them. Select `Configure elements, excluding them by default` from the drop-down menu 'Do you want to change the elements from the additional tagsets in any way?'.
4. Tick the check box 'Define some new elements'
5. When you have done this, press `Submit`
6. You will see a form allowing you to add a new element, like this:



. Unless you feel confident at working out what the possibilities are for yourself, we suggest that that you want:

(a) an element called 'soundClip'
(b) as a member of the class 'hqphrase'
(c) with the attribute class 'xPointer'
(d) with the content model 'TEXT'
(e) with some sort of description

When you are done, press `Submit`

7. You will see a list of all the elements now available for inclusion in your schema. Click on any element name to see full information about it. This will take you to the formal definition for that element within the TEI Guidelines. Use the Back button of your browser to return to the list of available tags. (If the page has expired from the cache, you may need to reload it). Explore the meanings and usage of any elements you are curious about.

8. Now look at the two radio buttons next to each element elements. If the first of these is selected, then that element will be included in your schema; if the second is selected (as they all currently are, except for headers), then that element will be excluded from your schema.

9. For this exercise, you need to *include* the following elements:

   from the figures module : `<figure>`, `<figDesc>`

   from the header module : all items are forced to be included, whatever choice you made earlier. Leave these as they are, as the TEI header is a savage beast when cornered

   from the structure module `<body>`, `<div>`,and `<text>`

   from the core module `<date>`, `<head>`, `<item>`, `<list>`, `<name>`, `<note>`, `<p>`, `<respStmt>`, and `<title>`

10. When you've finished, click on the `Submit` button. The application will now send you a RelaxNG compact schema. Depending on your browser, you should be asked if you want to save it. Choose the location where you unpacked the samples for today, and save the schema under the name `edison.rnc`. Look at the result, if you feel strong, or experiment with other options of the web application.

11. We've prepared a little test program which you can use to check that you've made your schema correctly: look in your folder, and you will see a file called `edison.xml`. Saving the schema with the same will allow Emacs to match it up to the XML file. Open the XML file with Emacs and look at the mode line—does it say Valid? If not, click on the word Invalid, and try to see what went wrong.

3 Making a real file

Your next challenge is to make an XML document which includes a picture and a soundclip, and then to transform it to HTML for display on a website.

We have provided a couple of example objects for your use in the sample directory, called edison.jpg and edison.mp3 respectively. One shows a photograph of Edison annotated by the great man himself: the story goes that this was found only slightly charred after a fire which destroyed Edison's original factory in New Jersey. The sound clip is the famous Mary had a little lamb recording, as recounted by Edison in a recording made in 1927.

With your new schema, you can now directly reference these objects in your document, using a `<figure>` and a `<soundClip>` element respectively. Here is what they might look like:

```
<p>...<figure url="edison.jpg">
    <figDesc>Photograph of Edison
     annotated by himself in <date>1887</date></figDesc></figure>
...<soundClip  url="edison.mp3">
      Edison reminisces about his first
      phonograph recording
      </soundClip>
     </p>
```

Once your document is valid, you can transform it into an HTML web page by using an XSLT stylesheet. We have prepared a suitable stylesheet for this purpose in the file `display.xsl`. You can invoke this by adding a a stylesheet reference like the following: `<?xml-stylesheet type="text/xsl" href="display.xsl"?>` at the start of your file. Some browsers (e.g. IE6) may render this XML directly, but a more reliable method is to run a free-standing XSLT processor such as saxon or xsltproc to generate a static HTML page from your document. Try typing

```
xsltproc -o edison.html display.xsl edison.xml
```

at the command prompt (Tools/Shell Command in Emacs). This will generate an HTML file called `edison.html` which any web browser should be able to display. What actually happens when the user clicks on the sound clip link will, of course, depend on how their browser is configured... but that is another story.

4 Making a TEI header

OK, deep breath. Now all you need to do is make a really useful header for your document. Using the auto-prompting features of Emacs to help you, and the information in section 6. The TEI Header below, add all the metadata you can think of.

5 Class catalogue

5.1 TEI Model classes

class.Incl groups empty elements which may appear at any point within a TEI text.

class.addrPart groups elements which may constitute a postal or other form of address.

class.agent groups elements which contain names of individuals or corporate bodies.

class.bibl groups elements containing a bibliographic description.

class.biblPart groups elements which can appear only within bibliographic citation elements.

class.binary elements which express binary values in feature structures.

class.boolean groups elements which express Boolean values in feature structures.

class.categorize groups elements which may be used inside catDesc and appear multiple times

class.chunk groups elements which can occur between, but not within, paragraphs and other chunks.

class.common groups common chunk- and inter-level elements.

class.comp.dictionaries groups those component-level elements which are unique to the base tag set for dictionaries.

class.comp.drama groups those component-level elements which are specific to performance texts.

class.comp.spoken groups those elements which appear at the component level in spoken texts only.

class.comp.verse groups component level elements unique to the base tag set for verse.

class.complexVal groups elements which express complex feature values in feature structures.

class.data groups phrase-level elements containing names, dates, numbers, measures, and similar data.

class.date groups elements containing a date specifications.

class.demographic groups elements describing demographic characteristics of the participants in a linguistic interaction.

class.dictionaryParts groups all elements defined specifically for dictionaries.

class.dictionaryTopLevel groups related parts of a dictionary entry forming a coherent subdivision, for example a particular sense, homonym, etc.

class.divbot groups elements which can occur at the end of a text division; for example, trailer, byline, etc.

class.divtop groups elements which can occur at the start of any division class element.

class.dramafront groups elements which appear at the level of divisions within front or back matter of performance texts only.

class.edit groups phrase-level elements for simple editorial correction and transcription.

class.editIncl groups empty elements which perform a specifically editorial function, for example by indicating the start of a span of text added, deleted, or missing in a source.

class.encoding groups elements which may be used inside encodingDesc and appear multiple times

class.featureVal groups elements which express feature values in feature structures.

class.fmchunk groups elements which can occur as direct constituents of front matter, when a full title page is not given.

class.formInfo groups elements allowed within a form element in a dictionary.

class.formPointers groups elements in the dictionary base which point at orthographic or pronunciation forms of the headword.

class.fragmentary groups elements which mark the beginning or ending of a fragmentary manuscript or other witness.

class.front groups elements which appear at the level of divisions within front or back matter.

class.gramInfo groups those elements allowed within a gramGrp element in a dictionary.

class.header groups elements which may be used inside teiHeader and appear multiple times

class.hqinter groups elements related to highlighting which can appear either within or between chunk-level elements.

class.hqphrase groups phrase-level elements related to highlighting.

class.inter groups elements of the intermediate (inter-level) class: these elements can occur both within and and between paragraphs or other chunk-level elements.

class.lists groups all list-like elements.

class.loc groups elements used for purposes of location and reference

class.metadata groups empty elements which describe the status of other elements, for example by holding groups of links or of abstract interpretations, or by providing indications of certainty etc., and which may appear at any point in a document.

class.morphInfo groups elements which provide morphological information within the dictionary tag set.

class.notes groups all note-like elements.

class.oddDecl groups elements which generate declarations in some markup language in ODD documents.

class.oddPhr groups ODD documentation elements which appear at the phrase level, such as identifiers, values, tags, examples etc.

class.oddRef groups elements which reference declarations in some markup language in ODD documents.

class.paragraph The paragraph element, make into a class for the purpose of interchange.

class.personPart groups those elements which form part of a personal name.

class.phrase groups those elements which can occur at the level of individual words or phrases.

class.phrase.verse groups phrase-level elements which may appear within verse only.

class.placePart groups those elements which form part of a place name.

class.profile groups elements which may be used inside profileDesc and appear multiple times

class.refsys groups milestone-style elements used to represent reference systems

class.seg groups elements used for arbitrary segmentation.

class.segment groups segmenting elements.

class.singleVal group elements which express single feature values in feature structures.

class.stageDirection groups elements containing specialized stage directions defined in the additional tag set for performance texts.

class.teiHeader Metadata elements at the top of a TEI document

class.teiText Main element covering the body of a TEI document

class.temporalExpr groups component elements of temporal expressions involving dates and time, and defines an additional set of attributes common to them.

class.tpParts groups those elements which can occur as direct constituents of a title page (docTitle, docAuth, docImprint, epigraph, etc.)

5.2 TEI Attribute classes

class.analysis default declaration for class analysis: when the additional tag set for simple analysis is not selected, no attributes are defined for this class.

class.declarable groups elements which may be independently selected (using the special purpose decls attribute) from a candidate list of declarations within a TEI header.

class.declaring groups elements which may be independently associated with a particular declarable element within the header, thus overriding the inherited default for that element.

class.dictionaries defines a set of global attributes available on elements in the base tag set for dictionaries.

class.divn defines a set of attributes common to all elements which behave in the same way as divisions.

class.enjamb groups elements bearing the enjamb attribute.

class.entries groups the different styles of dictionary entries.

class.formPointers groups elements in the dictionary base which point at orthographic or pronunciation forms of the headword.

class.fragmentary groups elements which mark the beginning or ending of a fragmentary manuscript or other witness.

class.global defines a set of attributes common to all elements in the TEI encoding scheme.

class.interpret defines the set of attributes common to this group of interpretative elements.

class.linking default declaration for class linking: when the additional tag set for linking is not selected, no attributes are defined for this class.

class.metrical defines a set of attributes which certain elements may use to represent metrical information.

class.names groups those elements which refer to named persons, places, organizations etc.

class.personPart groups those elements which form part of a personal name.

class.pointer defines a set of attributes used by all elements which point to other elements by means of one or more IDREF values.

class.pointerGroup defines a set of attributes common to all elements which enclose groups of pointer elements.

class.readings defines a set of attributes common to all elements representing variant readings in text critical work.

class.seg groups elements used for arbitrary segmentation.

class.temporalExpr groups component elements of temporal expressions involving dates and time, and defines an additional set of attributes common to them.

class.terminology default declaration for class terminology: when the base tag set for terminological data is not selected, no attributes are defined for this class.

class.timed defines a set of attributes common to those elements which have a duration in time, expressed either absolutely or by reference to an alignment map.

class.typed defines a set of attributes which can be used to classify or subclassify certain elements in any way.

class.xPointer defines a set of attributes used by all those elements which use the TEI extended pointer mechanism to point at locations which have neither an SGML nor an XML ID.

5.3 TEI patterns

macro.componentPlus defines a sequence of components as needed in the general base tag set, allowing components from any base to be used, but preventing their mixing.

macro.componentSeq defines a sequence of component-level elements (such as paragraphs or lists) which can occur directly within text divisions and in similar positions.

macro.paraContent defines the legal version for paragraphs and similar elements.

macro.phrasegroup defines a phrase as character data or any phrase-level element.

macro.specialPara defines the content model of elements such as notes or list items, which either contain a series of component-level elements or else have the same structure as a paragraph, containing a series of phrase-level and inter-level elements.

macro.phraseSeq defines a sequence of character data and phrase-level elements.

macro.component defines the set of component-level elements for prose; these are elements which can appear directly within text bodies or text divisions.

6 The TEI Header

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.

6.1 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>
```

6.1.1 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>
```

6.1.2 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.

6.1.3 The Extent Statement

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

Example:

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

6.1.4 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>
```

6.1.5 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.

6.1.6 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 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.

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>
```

6.2 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.

6.2.1 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>
```

6.2.2 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>
```

6.2.3 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 DIV and DIV 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 sub-components 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.

6.3 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>
```

6.4 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>
```