27 Tag Set Documentation

Some minor revisions have been made in the way that the tag set documentation is used in producing the current XML version of the Guidelines; these are not as yet reflected in the recommendations of the present chapter.

This chapter describes an auxiliary DTD which may be used for the documentation of new elements, element classes and entities. It is primarily intended for use by those wishing to extend or modify the content of these Guidelines in a conformant manner, as described in chapters 29 *Modifying and Customizing the TEI DTD* and 28 *Conformance*; it may also be useful for the documentation of other encoding schemes. The elements described here are those used to document the TEI scheme itself, in part VII of the current document.

Three distinct elements are used to document a tag set, the contents of each of which is described in more detail in the appropriate section of this chapter.

<tagDoc> documents the structure, content, and purpose of a single element type. Attributes include:

usage specifies the optionality of an attribute or element.

Legal values are:

`req` required
`mwa` mandatory when applicable
`rec` recommended
`rwa` recommended when applicable
`opt` optional

<classDoc> contains reference information for a TEI *element class*; that is a group of elements which appear together in content models, or which share some common attribute, or both. Attributes include:

type indicates whether this is a model class, an attribute class, or both.

Legal values are:

`model` members of this class appear in the same content models
`atts` members of this class share common attributes
`both` members of this class share attributes and also appear in the same content models

<entDoc> formally documents a single named entity used within an SGML or XML encoding scheme. Attributes include:

type indicates whether this is a general or a parameter entity.

Legal values are:

`pe` parameter entity
`ge` general entity

In addition to these documentary elements, the following phrase-level elements may be found useful for marking up occurrences of element names etc. within the body of running text.

<gi> contains the name (generic identifier) of an element. Attributes include:

tei indicates whether this element is part of the TEI encoding scheme (i.e. defined in a TEI DTD fragment) or not.

Legal values are:

`yes` this element is part of the TEI scheme.
`no` this element is not part of the TEI scheme.

<att> contains the name of an attribute appearing within running text. Attributes include:

tei indicates whether this attribute is part of the TEI scheme (i.e., defined in a TEI DTD fragment) or not.

Legal values are:

`yes` this attribute is part of the TEI scheme.
`no` this attribute is not part of the TEI scheme.

<val> contains a single attribute value.

<tag> contains text of a complete start- or end-tag, possibly including attribute specifications, but excluding the opening and closing markup delimiter characters. Attributes include:

TEI indicates whether this tag is valid within the TEI scheme or not.

Legal values are:

`yes` this is a valid TEI tag.

`no` this is not a valid TEI tag.

These four elements are included in the phrase-level elements available to any document using the auxiliary tag set defined in this chapter; to make them available to documents using other DTDs, an appropriate parameter entity should be defined.

As an example of the recommended use of these elements, we quote from an imaginary TEI working paper:

```
<p>The <gi>gi</gi> element is used to tag element names when
they appear in the text; the <gi>tag</gi> element however is
used to show how a tag as such might appear. So one might
talk of an occurrence of the <gi>blort</gi> element which had
been tagged <tag>blort type='runcible'</tag>. The <att>type</att>
attribute may take any name token as value; the default value is
<val>spqr</val>, in memory of its creator.</p>
```

These elements and their components make up the auxiliary DTD for tag documentation, which is contained in the file teitsd2.dtd. This file has the following overall structure:

```
<!-- 27.: File teitsd2.dtd:  Auxiliary DTD for Tag Set Documentation-->
<!--Text Encoding Initiative Consortium:
Guidelines for Electronic Text Encoding and Interchange.
Document TEI P4, 2002.
Copyright (c) 2002 TEI Consortium. Permission to copy in any form
is granted, provided this notice is included in all copies.
These materials may not be altered; modifications to these DTDs should
be performed only as specified by the Guidelines, for example in the
chapter entitled 'Modifying the TEI DTD'
These materials are subject to revision by the TEI Consortium. Current versions
are available from the Consortium website at http://www.tei-c.org-->
<!--Embed entities for TEI generic identifiers.-->
<!ENTITY % TEI.elementNames PUBLIC '-//TEI P4//ENTITIES Generic
Identifiers//EN' 'teigis2.ent' >%TEI.elementNames;
<!--Define entities for TEI keywords.-->
<!ENTITY % TEI.keywords.ent PUBLIC '-//TEI P4//ENTITIES TEI
Keywords//EN' 'teikey2.ent' >%TEI.keywords.ent;
<!--Define element classes for content models, shared
attributes for element classes, and global attributes.  (This all
happens within the file TEIclas2.ent.)-->
<!ENTITY % TEI.elementClasses PUBLIC '-//TEI P4//ENTITIES TEI
ElementClasses//EN' 'teiclas2.ent' >%TEI.elementClasses;
<!--Embed the core tag set-->
<!ENTITY % TEI.core.dtd PUBLIC '-//TEI P4//ELEMENTS Core Elements//EN'
'teicore2.dtd' >%TEI.core.dtd;
<!--Define the top-level element for this DTD-->
<!ELEMENT tsd %om.RO;  ((tagDoc | entDoc | classDoc)+)>
<!ATTLIST tsd
      %a.global;
      TEIform CDATA 'tsd'  >
<!--Define some additions for the phrase level tags-->
<!ELEMENT gi %om.RO; (#PCDATA)>
<!ATTLIST gi
      %a.global;
      tei (yes|no) "yes"
      TEIform CDATA 'gi'  >
<!ELEMENT tag %om.RR; (#PCDATA)>
<!ATTLIST tag
      %a.global;
      TEI ( yes | no ) "yes"
      TEIform CDATA 'tag'  >
<!ELEMENT att %om.RR; (#PCDATA)>
<!ATTLIST att
      %a.global;
      tei (yes|no) "yes"
```

```
      TEIform CDATA 'att'  >
<!ELEMENT val %om.RO; (#PCDATA)>
<!ATTLIST val
      %a.global;
      TEIform CDATA 'val'  >
<!--Finally we define the elements specific to this DTD-->
<!--declarations from 27.1: The TagDoc element inserted here -->
<!--declarations from 27.1.1: Attribute documentation inserted here -->
<!--declarations from 27.2: Element classes inserted here -->
<!--declarations from 27.3: Entity Documentation inserted here -->
<!-- end of 27.-->
```

27.1 The TagDoc Documentation Element

The <tagDoc> element is used to document an element type, together with its associated attributes. A completely specified <tagDoc> may comprise all of the following components in the order specified:

<rs> contains a general purpose name or referring string. Attributes include:

type indicates more specifically the object referred to by the referencing string. Values might include "person", "place", "ship", "element" etc.

Values Any string of characters.

<gi> contains the name (generic identifier) of an element. Attributes include:

tei indicates whether this element is part of the TEI encoding scheme (i.e. defined in a TEI DTD fragment) or not.

Legal values are:

`yes` this element is part of the TEI scheme.

`no` this element is not part of the TEI scheme.

<desc> contains a brief description of the purpose and application for an element, attribute, or attribute value.

<attList> contains documentation for all the attributes associated with this element, as a series of <attDef> elements.

<exemplum> contains a single example demonstrating the use of an element, together with optional paragraphs of commentary.

<eg> contains a single example demonstrating the use of an element or attribute.

<remarks> contains any commentary or discussion about the usage of an element, attribute, class, or entity not otherwise documented within the containing element.

<part> specifies the module or part to which a particular element, element class, or entity belongs in a modular encoding scheme such as the TEI. Attributes include:

type indicates whether the tag set is a base, additional, core, or auxiliary tag set.

Suggested values include:

`core` a core tag set (part of every document)

`base` a base tag set

`add` an additional tag set

`aux` an auxiliary tag set

name indicates the specific tag set or part in question, usually by means of an identifier or short form.

Values any string of characters

<classes> specifies all the classes of which the documented element or class is a member or subclass. Attributes include:

names lists the identifiers of all classes of which the documented element or class is a member or subclass, possibly using parentheses to indicate inheritance.

Values a list of class names separated by spaces or commas, and optionally enclosed by parentheses; each name should be the class name specified for some element class in the scheme being documented or modified.

<files> specifies the name of the operating system file(s) within which this markup component is declared. Attributes include:

names supplies the names of one or more files.
Values a file identifier

<dataDesc> specifies the legal content of the element being documented, noting any semantic or application-dependent constraints, as well as constraints enforced by the content model.

<parents> lists elements which can directly contain this element.

<children> lists the elements which this element may directly contain.

<elemDecl> contains the text of a declaration for the element documented.

<attlDecl> contains the ATTLIST declaration associated with this element.

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

target specifies the destination of the pointer by supplying the values used on the id attribute of one or more other elements in the current document
Values One or more valid identifiers, separated by white space.

<equiv> specifies an equivalent or comparable element in some other markup language. Attributes include:

scheme names the markup language or encoding scheme
Values any phrase identifying a markup language

As the content model for <tagDoc> makes clear, only the <gi>, <desc>, <exemplum>, <dataDesc>, <parents>, <children> and <elemDecl> elements are mandatory components. For elements bearing attributes, the <attList> and <attlDecl> components are also required for TEI conformance. For compatibility with the TEI system, use of the <classes> and <files> elements is strongly recommended. The only components of the <tagDoc> element which can appear more than once are the <exemplum>, <ptr> and <equiv> elements. The order of components may not be changed.

The <tagDoc> and its constituents are defined as follows:

```
<!-- 27.1: The TagDoc element-->
<!ELEMENT tagDoc %om.RR; (gi, rs?, desc, attList?, exemplum*, remarks?,
   part?, classes?, files?, dataDesc?, parents?, children?,
   elemDecl, attlDecl?, ptr*, equiv*)>
<!ATTLIST tagDoc
      %a.global;
      usage (req|mwa|rec|rwa|opt) "opt"
      TEIform CDATA 'tagDoc'  >
<!--RS and PTR are defined in the core-->
<!--GI is defined above -->
<!ELEMENT desc %om.RO; %paraContent;>
<!ATTLIST desc
      %a.global;
      TEIform CDATA 'desc'  >
<!ELEMENT attList %om.RO; (attDef*)>
<!ATTLIST attList
      %a.global;
      TEIform CDATA 'attList'  >
<!ELEMENT exemplum %om.RR; (p*, eg, p*)>
<!ATTLIST exemplum
      %a.global;
      TEIform CDATA 'exemplum'  >
<!ELEMENT eg %om.RR; (#PCDATA)>
<!ATTLIST eg
      %a.global;
      TEIform CDATA 'eg'  >
<!ELEMENT remarks %om.RO; (%component.seq;)>
<!ATTLIST remarks
      %a.global;
      TEIform CDATA 'remarks'  >
<!ELEMENT part %om.RO; (#PCDATA)>
<!ATTLIST part
      %a.global;
      type CDATA #IMPLIED
      name CDATA #IMPLIED
```

```
      TEIform CDATA 'part'  >
<!ELEMENT classes %om.RO; (#PCDATA)>
<!ATTLIST classes
      %a.global;
      names CDATA #REQUIRED
      TEIform CDATA 'classes'  >
<!ELEMENT files %om.RO; EMPTY>
<!ATTLIST files
      %a.global;
      names CDATA #IMPLIED
      TEIform CDATA 'files'  >
<!ELEMENT dataDesc %om.RO; %phrase.seq;>
<!ATTLIST dataDesc
      %a.global;
      TEIform CDATA 'dataDesc'  >
<!ELEMENT parents %om.RO; (#PCDATA)>
<!ATTLIST parents
      %a.global;
      TEIform CDATA 'parents'  >
<!ELEMENT children %om.RO; (#PCDATA)>
<!ATTLIST children
      %a.global;
      TEIform CDATA 'children'  >
<!ELEMENT elemDecl %om.RO; (#PCDATA)>
<!ATTLIST elemDecl
      %a.global;
      TEIform CDATA 'elemDecl'  >
<!ELEMENT attlDecl %om.RR; (#PCDATA)>
<!ATTLIST attlDecl
      %a.global;
      TEIform CDATA 'attlDecl'  >
<!ELEMENT equiv %om.RO; %specialPara;>
<!ATTLIST equiv
      %a.global;
      scheme CDATA #REQUIRED
      TEIform CDATA 'equiv'  >
<!-- end of 27.1-->
```

27.1.1 The AttList Documentation Element

The `<attList>` element is used to document information about a collection of attributes, either within a `<tagDoc>`, or within a `<classDoc>`. It consists of a series of `<attDef>` elements, each documenting a single attribute and each using an appropriate selection from the following elements:

<attDef> contains the definition of a single attribute. Attributes include:

usage specifies the optionality of an attribute or element.

Legal values are:

`req` required
`mwa` mandatory when applicable
`rec` recommended
`rwa` recommended when applicable
`opt` optional

<attName> contains the name of the attribute being defined by an `<attDef>` element.

<rs> contains a general purpose name or referring string. Attributes include:

type indicates more specifically the object referred to by the referencing string. Values might include “person”, “place”, “ship”, “element” etc.

Values Any string of characters.

<desc> contains a brief description of the purpose and application for an element, attribute, or attribute value.

<datatype> specifies the *declared value* for an attribute.

<valList> contains a list of value and description pairs for an attribute. Attributes include:

type specifies the extensibility of the list of attribute values specified.

Legal values are:

`closed` only the values specified are permitted.

`semi` all the values specified should be supported, but other values are legal and software should have appropriate fallback processing for them.

`open` the values specified are sample values only.

<val> contains a single attribute value.

<valDesc> specifies any semantic or syntactic constraint on the value that an attribute may take, additional to the information carried by the <datatype> element.

<default> specifies the default declared value for an attribute.

<eg> contains a single example demonstrating the use of an element or attribute.

<remarks> contains any commentary or discussion about the usage of an element, attribute, class, or entity not otherwise documented within the containing element.

<equiv> specifies an equivalent or comparable element in some other markup language. Attributes include:

scheme names the markup language or encoding scheme

Values any phrase identifying a markup language

It will be noted that several of these elements are used identically to document both elements and attributes. Specific to attributes are <dataType>, <valList>, <valDesc>, <val> and <default>. For any attribute documented in this way, either a <valList> or a <valDesc> must be supplied to specify the range of permitted values for an attribute. A <valList> should be used if the intended set of values can be enumerated; a <valDesc> if it cannot. A legal <attDef> specification must contain an <attName>, a <desc>, a <datatype>, either a <valList> or a <valDesc>, and a default; the other elements listed above are all optional.

The <attList> within a <tagDoc> is used to specify only the attributes which are specific to that particular element. Attributes which are shared by other elements, or by all elements, should be documented by an <attList> contained within a <classDoc> element, as described in section 27.2 *Element Classes* below.

The following <attList> demonstrates some of the possibilities; for more detailed examples, consult the tagged version of the reference material in these Guidelines.

```
<attDef usage='opt'>
<attName>type</attName>
<desc>describes the form of the list.</desc>
<datatype>CDATA</datatype>
<valList type='semi'>
    <val>ordered</val><desc>list items are numbered or lettered.</desc>
    <val>bulleted</val><desc>list items are marked with a
       <soCalled>bullet</soCalled> or other typographic device.</desc>
    <val>simple</val><desc>list items are not numbered or bulleted.</desc>
    <val>gloss</val><desc>each list item glosses some term or
      concept, which is given by a <gi>label</gi> element preceding
      the list item.</desc>
</valList>
<default>simple</default>
<remarks>
<p>The formal syntax of the element declarations allows
<gi>label</gi> tags to be omitted from lists tagged <tag>list
type=gloss</tag>; this is however a semantic error.</p></remarks>
</attDef>
```

Those elements from the above list which are unique to attributes are declared as follows:

```
<!-- 27.1.1: Attribute documentation-->
<!ELEMENT attDef %om.RO; (attName, rs?, desc,
(datatype, (valList | valDesc)?),
default, eg?, remarks?, equiv*)>
<!ATTLIST attDef
      %a.global;
```

```
      usage (req|mwa|rec|rwa|opt) "opt"
      TEIform CDATA 'attDef'  >
<!ELEMENT attName %om.RO; (#PCDATA) >
<!ATTLIST attName
      %a.global;
      TEIform CDATA 'attName'  >
<!ELEMENT datatype %om.RO; (#PCDATA)>
<!ATTLIST datatype
      %a.global;
      TEIform CDATA 'datatype'  >
<!ELEMENT valList %om.RR; ((val,desc)*)>
<!ATTLIST valList
      %a.global;
      type (closed | semi | open) "open"
      TEIform CDATA 'valList'  >
<!ELEMENT valDesc %om.RO; %phrase.seq;>
<!ATTLIST valDesc
      %a.global;
      TEIform CDATA 'valDesc'  >
<!ELEMENT default %om.RO; (#PCDATA) >
<!ATTLIST default
      %a.global;
      TEIform CDATA 'default'  >
<!-- end of 27.1.1-->
```

27.2 Element Classes

The element `<classDoc>` is used to document an *element class*, as defined in section 3.7 *Element Classes*. It has the following components:

<classDoc> contains reference information for a TEI *element class*; that is a group of elements which appear together in content models, or which share some common attribute, or both. Attributes include:

type indicates whether this is a model class, an attribute class, or both.

Legal values are:

`model` members of this class appear in the same content models

`atts` members of this class share common attributes

`both` members of this class share attributes and also appear in the same content models

<class> specifies the name of an element class.

<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.

Values Any string of characters.

<desc> contains a brief description of the purpose and application for an element, attribute, or attribute value.

<attList> contains documentation for all the attributes associated with this element, as a series of `<attDef>` elements.

<remarks> contains any commentary or discussion about the usage of an element, attribute, class, or entity not otherwise documented within the containing element.

<part> specifies the module or part to which a particular element, element class, or entity belongs in a modular encoding scheme such as the TEI. Attributes include:

type indicates whether the tag set is a base, additional, core, or auxiliary tag set.

Suggested values include:

`core` a core tag set (part of every document)

`base` a base tag set

`add` an additional tag set

`aux` an auxiliary tag set

name indicates the specific tag set or part in question, usually by means of an identifier or short form.
Values any string of characters

<classes> specifies all the classes of which the documented element or class is a member or subclass. Attributes include:

names lists the identifiers of all classes of which the documented element or class is a member or subclass, possibly using parentheses to indicate inheritance.
Values a list of class names separated by spaces or commas, and optionally enclosed by parentheses; each name should be the class name specified for some element class in the scheme being documented or modified.

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

target specifies the destination of the pointer by supplying the values used on the id attribute of one or more other elements in the current document
Values One or more valid identifiers, separated by white space.

<equiv> specifies an equivalent or comparable element in some other markup language. Attributes include:

scheme names the markup language or encoding scheme
Values any phrase identifying a markup language

Of these elements, only the <class> and <desc> elements are required components. If present, the other elements must be given in the order specified, and only <ptr> and <equiv> may be repeated.

The attribute type is used to distinguish between 'model' and 'attribute' classes; for the former, a <classDoc> simply exists so that members of the class it documents may point to it (by specifying the value of its id attribute among the values specified by the names attribute of their <classes> component); for the latter, the <classDoc> additionally contains an <attList> which specifies the attributes shared by the members of the class. A class may perform both functions, of course.

Where a class inherits properties or attributes from some other class, the <classes> element may be used to indicate this fact. Membership of an attribute class can be inherited by any class, but model-only classes may not include attribute-only classes amongst their members. For further discussion of the TEI class system, see section 3.7 *Element Classes*.

The <classDoc> element and the elements unique to it are declared as follows:

```
<!-- 27.2: Element classes-->
<!ELEMENT classDoc %om.RO; (class, rs?, desc, attList?, remarks?, part?,
classes?, files?, ptr*, equiv*) >
<!ATTLIST classDoc
      %a.global;
      type (model | atts | both) #IMPLIED
      TEIform CDATA 'classDoc'  >
<!ELEMENT class %om.RO; (#PCDATA)>
<!ATTLIST class
      %a.global;
      TEIform CDATA 'class'  >
<!--all other constituents are defined above-->
<!-- end of 27.2-->
```

27.3 Entity Documentation

The <entDoc> element is used to document any other entity not otherwise documented by the elements described in this chapter. Its chief uses are to provide systematic documentation for parameter entities used within TEI DTD fragments (for example, those used to enable different components of the TEI DTD, or to describe common content models), but it may be used for any purpose. It has the following components:

<entDoc> formally documents a single named entity used within an SGML or XML encoding scheme. Attributes include:

type indicates whether this is a general or a parameter entity.
Legal values are:
pe parameter entity
ge general entity

<entName> contains the full name of an entity, excluding the percent sign in the case of a parameter entity.

<rs> contains a general purpose name or referring string. Attributes include:
type indicates more specifically the object referred to by the referencing string. Values might include “person”, “place”, “ship”, “element” etc.
Values Any string of characters.

<desc> contains a brief description of the purpose and application for an element, attribute, or attribute value.

<remarks> contains any commentary or discussion about the usage of an element, attribute, class, or entity not otherwise documented within the containing element.

<string> contains the intended expansion for the entity documented by an <entDoc> element, enclosed by quotation marks.

<ptr> defines a pointer to another location in the current document in terms of one or more identifiable elements. Attributes include:
target specifies the destination of the pointer by supplying the values used on the id attribute of one or more other elements in the current document
Values One or more valid identifiers, separated by white space.

<equiv> specifies an equivalent or comparable element in some other markup language. Attributes include:
scheme names the markup language or encoding scheme
Values any phrase identifying a markup language

Of these, only <entName>, <desc> and <string> are required components. If present, the other elements must be given in the order specified, and only <ptr> and <equiv> may be repeated.

The <entDoc> element and the elements unique to it are declared as follows:

```
<!-- 27.3: Entity Documentation-->
<!ELEMENT entDoc %om.RR; (entName, rs?, desc, remarks?, string, ptr*,
equiv*)>
<!ATTLIST entDoc
      %a.global;
      type (pe | ge) #REQUIRED
      TEIform CDATA 'entDoc'  >
<!ELEMENT entName %om.RO; (#PCDATA)>
<!ATTLIST entName
      %a.global;
      TEIform CDATA 'entName'  >
<!ELEMENT string %om.RR; (#PCDATA)>
<!ATTLIST string
      %a.global;
      TEIform CDATA 'string'  >
<!--All other constituents are defined above-->
<!-- end of 27.3-->
```

VI: Technical Topics