Telecommunications
Industry Forum

Sponsored by the Alliance for
Telecommunications Industry Solutions

Telecommunications Interchange Markup, Part 2: TIM DTD Reference

TIM Version 2

TCIF Guideline
TCIF-IPI-95-004-PART2
Issue 1A, June 1997

DRAFT FOR BALLOT

Telecommunications Interchange Markup, Part 2: TIM DTD Reference

Prepared by the TCIF Information Products Interchange Subcommittee, http://www.atis.org/atis/tcif/.

Prepared for TCIF by the Information Products Interchange Committee. For more information about TCIF, go to http://www.atis.org/atis/tcif/. To order this document, please contact TCIF at (202) 434-8836, FAX (202) 393-5453.

If you have questions or comments about this document, please contact Donald F. Pratt, (908) 699-4012, dfp@ims.bellcore.com.

About This Guideline

This is part 2 of TCIF's guide to the TIM DTD. Part 2 contains an alphabetized list of elements, attributes, standard attribute values, notations, entities, and keywords that occur in the DTD (including the TCIF version of the CALS Table DTD). The DTD itself could be reconstructed from the individual entries, but it is much better to simply download the DTD from a TCIF repository:

- **http://www.atis.org/atis/tcif/ipi**
- **ftp://ftp.bellcore.com/pub/world/TCIF** (command-line ftp only)
- **ftp://ftp.isogen.com/pub/tcif** (accessible with Web browsers)

Part 2 is meant for those who have already read Part 1, the introduction to TIM. It is the experienced user's reference. Both parts, and related TCIF documents, are available at the repository sites.

How the Reference is Organized

Every main entry and cross-reference entry in this reference is alphabetical. There are no entries for numerals and only one cross-reference entry for a symbol (#). There is no table of contents or index, because they would be redundant.

Types of entries

<Admon> indicates a main entry for an element in the TIM or TCIF CALS Table DTD. It is defined in an element declaration, which begins with ("<!ELEMENT").

alerts= indicates a main entry for an attribute. It is defined in the attribute-list declaration for at least one element, which begins with ("<!ATTLIST").

"Abstract", type= indicates a cross-reference for a standard value of an attribute. It has a link to the attribute entry. Some values are explicitly provided in the DTD as choices, but others are standard only by agreement.

AVI (notation) indicates a main entry for the keyword for a notation (a data format). It is defined in a notationdeclaration ("<!NOTATION")

%baseatts; indicates a main entry for a parameter entity reference. Content defined for it in an entity declaration ("<!ENTITY") is substituted for it wherever it appears in the DTD. Some parameter entities are defined to be a number of regular entities, like © which will be substituted for in a document instance rather than a DTD.

?ATTLINK indicates a main entry for a processing instruction. A processing instruction generally applies to only one application, but the instructions for HyTime, XML, and SoftQuad Panorama are included in the DTD because they are assumed to be of value to most users. Processing instructions begin with "<?".

footnote (no special characters) indicates a cross-reference for a term you might have expected to be in the DTD. It has a link to the appropriate entry.

A (Abs — AVI)

"About", type= — Section (S.4)

<Abs> A.1

May occur once within **DocID** (D.27) to provide a simple abstract of the document (suitable for a text-only on-line catalog).

```
<!ELEMENT Abs - - (#PCDATA)* >
<!ATTLIST Abs
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Specific forms for the information it contains are left to the discretion of the document originator. There is no length limit. The abstract will be read as a single paragraph, and tabs and line breaks will be treated as spaces. There is no provision for subelements or any kind of formatting within the abstract. A more richly formatted abstract, if provided, should be contained in a **Section** (**type="Abstract"**) within the **FrontMatter**.

"Abstract", type= — Section (S.4)

"Acronym", type= — ListTerm (L.14) and T **(T.1)**

"Action", type= — LI (L.8)

<Admon> A.2

Any admonishment (warning message) other than **Danger** (D.1), **Warning** (W.1), or **Caution** (C.3), which have established meanings in telecom documents.

```
<!ELEMENT Admon - - (Title?,(P|S|Annote|Frame|Table|OrderedList
          |UnorderedList|VariableList|SegmentedList|Listing|Pt|IndexTerm
          |Graphic|Object|Flowchart|ExternalText)*)>
<!ATTLIST Admon
          %baseatts;
          %identity;
          %numbering;
          %commonatts; >
```

Values of the **type** attribute that should be recognized: "Note", "Important", and "Attention" (these types overlap those of **Annote**, A.9); "Policy", "Safety", "Equipment" (equipment damage), and "Service" (service interruption). See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), and **%commonatts;** (C.19) for more on this element's attributes.

It is expected that the meanings, uses, and presentation specifications of admonishments will be the subject of a separate TCIF Guideline.

"alert", present= (P.16)

alerts= A.3

The **alerts** attribute should be used only for IDREFS (I.5) to elements with **present="alert"** (P.16). The element pointed to by the **alerts** attribute should be within the same document and typically would be an admonishment. Eventually, browsers and viewers may be able to present any elements linked to visible elements this way in their own pop-up windows, but until that has become a taken-for-granted feature, it is also necessary to place alerts immediately before elements referring to them, so that they will not be overlooked in simpler presentations. See **Caution** (C.3) for an example.

```
alerts IDREFS #IMPLIED
```

This attribute is defined for all elements that have the **%commonatts;** (C.19).

align= A.4

Applies to **TGroup**, **ColSpec**, **SpanSpec**, **Entry**, and **EntryTbl**. Specifies horizontal alignment of graphics and of text in cells. Values are inherited from higher-level elements. The value "Char" means alignment is on a specific character, identified by the **char** attribute (C.8). For **Frame**, **Graphic**, and **Object** the definition is:

```
align (left|right|center) #IMPLIED
```

For **TGroup** the definition is:

```
align (left|right|center|justify|char) "left"
```

For **SpanSpec** the definition is:

```
align (left|right|center|justify|char) "center"
```

For other elements the definition is:

```
align (left|right|center|justify|char) #IMPLIED
```

For table elements, the value will normally be inherited from **TGroup**.

"all", frame= (F.9)

<AltNo> A.5

May occur any number of times within **DocID** (D.27) to provide an alternative document number. For instance, a document may be assigned a part number within a product line as well as a document number within a publications catalog.

```
<!ELEMENT AltNo - - (#PCDATA)* >
```

```
<!ATTLIST AltNo
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Content and format are left to the document provider.

Example 1:

```
<AltNo>61635A</AltNo>
```

Example 2:

```
<AltNo>Part # 61635A</AltNo>
```

altreps= A.6

ID reference(s) to one or more alternative representations of an element's content (such as graphic images or text in another markup language). The referenced element should have its **`present`** attribute (P.16) set to "altrep".

```
altreps IDREFS #IMPLIED
```

Default value: null (meaning that there is no alternative representation for the element).

This attribute is defined for all elements that have the **`%commonatts;`** (C.19).

Style Note

An element using the **`altreps`** attribute must contain its own representation of the content (this is a change from TIM 1). Its own representation is the preferred one, to be presented by default, and an on-line viewer would probably relegate the contents of the alternative representations to pop-up windows accessed by clicking on icons.

anchrole= A.7

Anchor roles for **`Link`** (L.9). The HyTime standard requires that a role be assigned for each link end when more than one is specified. Because TIM defines only a general-purpose link, the anchor roles have uninteresting names:

```
anchrole NMTOKENS #FIXED "from to"
```

Animator (notation) A.8

Notation that should be declared for a `filename.fli` or `filename.flc` (AutoDesk Animator format) file.

```
<!NOTATION Animator  SYSTEM "Animator">
```

Example:

```
<!ENTITY scene001 SYSTEM "../other/scene001.fli" NDATA
Animator>

...

<Object entityref="scene001">
```

The file should be stored in an `other` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM document. Animator files may be included in TEDD packages experimentally. Preferred formats will be specified in future TCIF Guidelines. Currently the options for video files are Animator (no sound), AVI (A.17), MOV (M.7), and MPEG (M.8).

<Annote> A.9

Annotation: the various types are distinguished by the **`type`** attribute (T.23). The content model is that of an **`S`** structure (S.1). The **`present`** attribute should be set to "float" for an **`Annote`** that floats, as footnotes normally do.

```
<!ELEMENT Annote - - (Title?,(P|S|Annote|Frame|Danger|Caution|Warning
          |Admon|Table|OrderedList|UnorderedList|VariableList
          |SegmentedList|Listing|Pt|IndexTerm|Graphic|Object|Flowchart
          |ExternalText)*) >
<!ATTLIST Annote
          %baseatts;
          %identity;
          %numbering;
          %commonatts; >
```

See the entries for the entities **`%baseatts;`** (B.2), **`%identity;`** (I.3), **`%numbering;`** (N.13), and **`%commonatts;`** (C.19) for all this element's attributes.

Values of the **`type`** attribute that should be recognized: "Note", "Important", "Attention" (these types overlap those of **`Admon`**, A.2); "FNote" (floating note/ footnote); "AuthorNote", "EditorNote", and "PublisherNote"; "Comment", "ReviewerComment", and "ReaderComment".

Annotations, even when labeled "Important" or "Attention," should never be used to warn of risk to persons, equipment, or continuity of service: admonishments (specifically **`Danger`**, **`Caution`**, and **`Warning`**) should always be used in those situations.

Possible markup for a footnote:

```
... here is where the footnote occurs<Ref rid=XYZ><Sup>1</
Sup></Ref><Float><Annote type="FNote" present="float" id=XYZ
numeration=arabic suffix="." label="1."><P>This is the
footnote</P></Annote></Float>. Continuation of the para...
```

Comment

There is no consensus on the best way to mark up cross-references. See **Ref** (R.2).

"Appendix", type= — Section (S.4)

application= A.10

The application that is needed to show or play an **Object** (O.1).

```
application CDATA #IMPLIED
```

<AppMatter> A.11

An optional major subdivision of **TDoc** (T.9), containing appendixes (**<Section type="Appendix" ...>**). It exists, along with the other major subdivisions (optional **FrontMatter**, required **Body**, and optional **BackMatter**) mainly to aid applications in automatically identifying and enumerating the sections within it. The AppMatter is typically broken down into sections, each child section representing a separate appendix to the document.

```
<!ELEMENT AppMatter - - ((Title|TitleGroup)?,(Contents|Index|P|S|Annote
          |Frame|Danger|Caution|Warning|Admon|DistOrdLI|DistVarLI|Table
          |OrderedList|UnorderedList|VariableList|SegmentedList|Listing
          |Pt|IndexTerm|Graphic|Object|Flowchart|ExternalText
          |Section)*) >
<!ATTLIST AppMatter
          %baseatts;
          %identity;
          alerts IDREFS #IMPLIED
          altreps IDREFS #IMPLIED >
```

See the entries for the entities **%baseatts;** (B.2) and **%identity;** (I.3) for details on this element's attributes. Use of the **type** attribute for **AppMatter** is not currently supported (no standard values have been established).

arrange= A.12

How items of a **SimpleList** are arranged: "inline" (sentence form), "vert" (vertical, in columns with the second item below the first), or "horiz" (horizontal, in columns with the second item to the right of the first). See the example at **SimpleList** (S.15).

```
arrange (inline|vert|horiz) #IMPLIED
```

The default is "vert".

"ArticleTitle", type= — T (T.1)

"Attachments", type= — Contents (C.22)

"Attention", type= — Annote (A.9)

?ATTLINK A.13

Introduces a processing instruction needed for SoftQuad Panorama identifying the attribute **`url`** as one providing a link.

```
<?ATTLINK URLRef url URI>
```

ATTLIST (SGML reserved name A.14

Introduces the list of attributes for each element.

"Attribute", type= — T (T.1)

<Au> A.15

May occur any number of times within **`DocID`** (D.27) or **`Version`** (V.7) to provide information about the author(s) of the document.

```
<!ELEMENT Abs - - (#PCDATA)* >
<!ATTLIST Abs
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Multiple **`Au`** elements should be used for multiple authors. Specific uses of this element and specific forms for the information it contains are left to the discretion of the document provider. Example:

```
<Au>Fred Waxbean, Bellcore, (201) 829-0000</Au>
```

AU (notation) A.16

Notation that should be declared for a `filename.au` audio file.

```
<!NOTATION AU        SYSTEM "AU">
```

Example of use:

```
<!ENTITY clip001 SYSTEM "../au/clip001.au" NDATA AU>

...

<Object entityref="clip001">
```

The file should be stored in an `au` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s). AU files may be included in TEDD packages experimentally. Preferred formats will be specified in future TCIF Guidelines. Currently the options for sound files are AU, MIDI (M.5), and WAV (W.2).

"Audience", type= — Section (S.4)

"Author", type= — T (T.1)

An author index has **`type="Name"`**.

"AuthorNote", type= — Annote (A.9)

AVI (notation) A.17

Notation that should be declared for a `filename.avi` video file.

```
<!NOTATION AVI       SYSTEM "AVI">
```

Example:

```
<!ENTITY clip001 SYSTEM "../avi/clip001.avi" NDATA AVI>
...
<Object entityref="clip001">
```

The file should be stored in an `avi` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM document. AVI files may be included in TEDD packages experimentally. Preferred formats will be specified in future TCIF Guidelines. Currently the options for video files are Animator (no sound, A.8), AVI, MOV (M.7), and MPEG (M.8).

B (BackMatter — Body)

<BackMatter> B.1

An optional major subdivision of **TDoc** (T.9), containing any sections or indexes after the body sections and appendixes. It exists, along with the other major subdivisions (optional **FrontMatter**, required **Body**, and optional **AppMatter**) mainly to aid applications in automatically identifying and naming or enumerating the sections within it.

```
<!ELEMENT BackMatter - - ((Title|TitleGroup)?,(Contents|Index|P|S|Annote
          |Frame|Danger|Caution|Warning|Admon|DistOrdLI|DistVarLI|Table
          |OrderedList|UnorderedList|VariableList|SegmentedList|Listing
          |Pt|IndexTerm|Graphic|Object|Flowchart|ExternalText
          |Section)*) >
<!ATTLIST BackMatter
          %baseatts;
          %identity;
          alerts IDREFS #IMPLIED
          altreps IDREFS #IMPLIED >
```

See the entries for the entities **%baseatts;** (B.2) and **%identity;** (I.3) for details on this element's attributes. Use of the **type** attribute for **BackMatter** is not currently supported (no standard values have been established).

base (HyTime module) — HyTime (H.4)

%baseatts; B.2

An entity defining basic attributes available for almost all elements (except those in the **DocID**):

```
<!ENTITY % baseatts
         "type NMTOKENS #IMPLIED
          dstype CDATA #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          group NMTOKENS #IMPLIED
          status CDATA #IMPLIED
          version IDREFS #IMPLIED
          revstat (norev|revised|revflag|insert|insflag|delete|strike)
            #IMPLIED
          lang CDATA #IMPLIED
          present (normal|float|alert|altrep) #IMPLIED
          presspec CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED" >
```

Whenever "%baseatts;" appears in an element's ATTLIST, the thirteen attribute declarations above are substituted for it. Refer to the individual entries for the meanings and uses of each: **type** (T.23), **dstype** (D.33), **remap** (R.5), **remapatt** (R.10), **remapto** (R.12), **group** (G.4), **status** (S.23), **version**

(V.8), **revstat** (R.16), **lang** (L.3), **present** (P.16), **presspec** (P.17), and **TIM2** (T.19).

"Bibliography", type= — Section (S.4)

BMP (notation) B.3

Notation that should be declared for a `filename.bmp` bitmap graphic file (Windows or OS/2 Bitmap).

```
<!NOTATION BMP       PUBLIC "+//ISBN 0-7923-9432-1::Graphic
           Notation//NOTATION Microsoft Windows bitmap//EN">
```

Example of use:

```
<!ENTITY fig001 SYSTEM "../bmp/fig001.bmp" NDATA BMP>
```

BMP files should not be included in TEDD packages that follow all TCIF Guidelines. Currently the TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG (P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18).

<Body> B.4

A required major subdivision of **TDoc** (T.9). It exists, along with the three optional major subdivisions (**FrontMatter**, **AppMatter**, and **BackMatter**) mainly to aid applications in automatically identifying and numbering the sections within it.

```
<!ELEMENT Body - - ((Title|TitleGroup)?,(Contents|Index|P|S|Annote
          |Frame|Danger|Caution|Warning|Admon|DistOrdLI|DistVarLI|Table
          |OrderedList|UnorderedList|VariableList|SegmentedList|Listing
          |Pt|IndexTerm|Graphic|Object|Flowchart|ExternalText
          |Section)*) >
<!ATTLIST Body
          %baseatts;
          %identity;
          alerts IDREFS #IMPLIED
          altreps IDREFS #IMPLIED >
```

See the entries for the entities **%baseatts;** (B.2) and **%identity;** (I.3) for details on this element's attributes. Use of the **type** attribute for **Body** is not currently supported (no standard values have been established).

"bold", emph= (E.6)

"boldital", emph= (E.6)

"bottom", valign= (V.1)

C (catres — CURRENT)

catres= C.1

Concatenate results. Potentially used by a **DataLoc** (D.2) that retrieves data for processing rather than simply jumps to a location. Determines whether or not the results of a **DataLoc** link are concatenated. A value of "catres" means that all resulting data strings are concatenated. A value of "nocatres" means that each resulting data string is displayed or processed separately. A value of "catressp" means that a single white space is place between the items concatenated. If the links were to segments of a video, "catres" would probably be interpreted as "play the segments spliced together," while "catressp" would probably mean "play the segments in order, pausing between them."

```
catres (catres|catressp|nocatres) "nocatres">
```

"catressp", catres= (C.1)

catsrc= C.2

Concatenate sources. Potentially used by a **DataLoc** (D.2). Allows the concatenation of multiple source elements before the tokens are counted. A value of "catsrc" allows the data string to be counted across the concatenated elements as though they were one contiguous element. A value of "nocatsrc" requires that the data string be recounted in each separate source element. A value of "catsrcsp" causes the source concatenation to place a single white space between each of the source components. For instance, if the **NameLoc** (N.3) and/or **TreeLoc** (T.22) specifications of the link resolve to three separate elements and the **DataLoc** specifies the 10th through 20th words, "catsrc" would require concatenating the three elements before counting off the words, while "nocatsrc" would mean retrieving the 10th through 20th words from each of the three elements. (That could make sense in a link that retrieves data for processing, not one that jumps to a location.)

```
catsrc (catsrc|catsrcsp|nocatsrc) "nocatsrc"
```

"catsrcsp", catsrc= (C.2)

<Caution> C.3

An admonishment that warns of the risk of possible service interruption or of minor injury. Admonishments — the other types are **Admon** (A.2), **Danger** (D.1) and **Warning** (W.1) — are important in telecom documents, and ideally they should be given sophisticated presentation: they generally apply to a span of text, and should appear automatically whenever any part of that text is presented. They have the data-content model of an **S** structure (S.1).

```
<!ELEMENT Caution - - (Title?,(P|S|Annote|Frame|Table|OrderedList
```

```
          |UnorderedList|VariableList|SegmentedList|Listing|Pt|IndexTerm
          |Graphic|Object|Flowchart|ExternalText)*) >
<!ATTLIST Caution
          %baseatts;
          %identity;
          %numbering;
          %commonatts;>
```

See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), and **%commonatts;** (C.19) for all this element's attributes. Use of the **type** attribute is not supported.

It is expected that the meanings, uses, and presentation specifications of admonishments will be the subject of a separate TCIF Guideline.

Style Note

An admonishment that needs to be displayed along with a range of text (such as a step in a procedure) should be have its **present** attribute set to "alert" and be referenced by the **alerts** IDREFS attribute of the element(s) containing that range of text. But the admonishment should appear in the text stream immediately before the beginning of the range, in case the rendering application isn't capable of, or the medium doesn't allow, floating the admonishment and displaying it simultaneously with the elements it applies to.

Example 1:

```
<Caution id="TCIF-IPI-95-004-CAUTION-4"
present="alert"><P>Admonishments should be placed immediately
before the text they refer to in a TIM data stream, in case
the rendering application cannot float the admonishment and
display it simultaneously with the text</P></Caution>

<S alerts="TCIF-IPI-95-004-CAUTION-4"><P>Here's how
admonishments work:</P>

....

<P>And that is the whole story</P></S>
```

Example 2:

```
... </LI>

<Alert><Caution id="TCIF-IPI-95-004-CAUTION-4"><P>Make sure
the green ground wire is connected to Terminal B while
performing steps 2 and 3.</P></Caution></Alert>

<LI label="Step 2." alerts="TCIF-IPI-95-004-CAUTION-4"><P> ...

<LI label="Step 3." alerts="TCIF-IPI-95-004-CAUTION-4"><P>
```

CCITT-G3 (notation, TIM 1 only)

Currently the TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG

(P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18)

CDATA (SGML reserved name) C.4

Reserved name for text data containing no markup. Unlike PCDATA (P.3), SGML markup within CDATA is not recognized by the parser. In other words, all character strings are treated like regular text. Take, for example, the following text string:

```
TIRKS&reg; is a registered trademark of Bellcore<Ref rid="TCIF-
IPI-95-004-FNOTE-1"><Sup>1</Sup></Ref> ....
```

In PCDATA or RCDATA (R.1) but not CDATA, the "®" character entity reference would be replaced with the appropriate character (®) by the rendering application. In PCDATA, the subelements would be recognized, but in RCDATA or CDATA they would be just more text characters.

CDATA is used as the value type of many attributes in TIM. These attributes can have values that contain character entity references, and they will be recognized as if the type were RCDATA. Thus **<LI label="•">** will produce a bulleted list item.

Style Note

Character entity references in attribute values may cause problems for rendering applications. It is recommended that an attribute value contain either a single entity reference and nothing else, or simple character data without entity references.

<CDocType> C.5

An optional subelement of **DocID** (D.27), describing the type of "canonical document" provided, if any. The "canonical document" is an electronic file that exactly matches the printed copy or, even if not, is considered the official version of the document. Likely values are "PDF" and "PS", for Acrobat and PostScript files respectively.

```
<!ELEMENT CDocType - - (#PCDATA)* >
<!ATTLIST CDocType
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

CDverbatim (notation, TIM 1 only) — CDATA (C.4)

"center", align= (A.4)

"centered", presspec= (P.17)

CGM (notation) C.6

Notation that should be declared for a `filename.cgm` binary-encoded Computer Graphic Metafile (metafiles may contain drawings, bitmaps, and text).

```
<!NOTATION CGM       SYSTEM "CGM-BINARY">
```

Example of use:

```
<!ENTITY fig001 SYSTEM "../cgm/fig001.cgb" NDATA CGM-BINARY>
```

The file should be stored in a `cgm` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM document. CGM-BINARY files may be included in TEDD packages experimentally. Preferred formats for vector graphics and/or metafiles will be specified in a future TCIF Guideline. Currently the options for vector-graphic files and metafiles are CGM (C.6), CGM-CLEAR (C.7), DXF (D.35), EPS (E.12), PDF (P.5), PICT (P.9), WMF (W.5), and WPG (W.6). CGM-CHAR was a notation in TIM 1 for character-encoded CGM, which is considered obsolete.

CGM-BINARY (notation, TIM 1 only)

Renamed CGM (C.6) in TIM 2.

CGM-CHAR (notation, TIM 1 only) — CGM (C.6)

CGM-CLEAR (notation) C.7

Notation that should be declared for a `filename.cgm` clear-text-encoded Computer Graphic Metafile (metafiles may contain drawings, bitmaps, and text):

```
<!NOTATION CGM-CLEAR PUBLIC "ISO 8632/4//NOTATION Clear text
          encoding//EN">
```

The file should be stored in a `cgm` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM document. CGM-CLEAR files may be included in TEDD packages experimentally. Preferred formats will be specified in a future TCIF Guideline. Currently the options for vector-graphic files and metafiles are CGM (C.6), CGM-CLEAR (C.7), DXF (D.35), EPS (E.12), PDF (P.5), PICT (P.9), WMF (W.5), and WPG (W.6). CGM-CHAR was a notation in TIM 1 for character-encoded CGM, which is considered obsolete.

"Changes", type= — Section (S.4)

"Chapter", type= — Section (S.4)

char= C.8

Applies to the CALS Table elements **TGroup**, **ColSpec**, **SpanSpec**, **Entry**, and **EntryTbl**. Specifies a character on which to align when the **align** attribute has the value "**Char**". Usually this would be a decimal point, but it could be any character. **Charoff** (C.9) specifies where within the column to align the character. Values are inherited from higher-level elements. For **TGroup** the definition is:

```
char CDATA ""
```

For other elements the definition is:

```
char CDATA #IMPLIED
```

If no character is specified, all characters in the cell are aligned to the left of the point specified by **charoff**.

charoff= C.9

Character Offset. Applies to the CALS Table elements **TGroup**, **ColSpec**, **SpanSpec**, **Entry**, and **EntryTbl**. Specifies where in the horizontal space of a column the **char** specified for character alignment (previous attribute) is to be. The value is read as a percentage of column width. Values are inherited from higher-level elements. For **TGroup** the definition is:

```
charoff NUTOKEN "50"
```

For other elements the definition is:

```
charoff NUTOKEN #IMPLIED
```

The default value is "50".

"Citation", type= — T (T.1)

<Class> C.10

The **Class** element contains the class identifier of the TEDD (Telecommunications Electronic Document Delivery) package containing the **TDoc** instance:

```
<Class>TEDD-Document-3.0</Class>
```

Currently the **Class** element must match the example above exactly. The number 3 represents the third major iteration of the TEDD specification.

```
<!ELEMENT Class    - -  (#PCDATA)* >
<!ATTLIST Class
```

```
id ID #IMPLIED
remap NMTOKEN #IMPLIED
remapatt CDATA #IMPLIED
remapto CDATA #IMPLIED
TIM2 NMTOKEN #IMPLIED >
```

It is a required component of the **DocID** (D.27).

close= C.11

Identifies an element closed at the point specified. A **Pt** element with **close** specified would normally be paired with one having **remap** specified, so that the pair would mark the beginning an end of an element in an alternative structure, as described under **Pt** (P.23).

```
close CDATA #IMPLIED
```

"Code", type= — Listing (L.12), T (T.1)

colname= C.12

Applies to the CALS Table elements **ColSpec**, **Entry**, and **EntryTbl**. Specifies a name for a column, which is needed for **SpanSpec** to indicate which columns a cell spans. Values are inherited from higher-level elements.

```
colname NMTOKEN #IMPLIED
```

The default value is undefined.

colnum= C.13

Applies to the CALS Table element **ColSpec**. Specifies the sequential number of the column from left to right.

```
colnum NMTOKEN #IMPLIED
```

The default value is calculated from the number of previous occurrences of **ColSpec**.

cols= C.14

Applies to the CALS Table elements **TGroup** and **EntryTbl**. Specifies the number of columns.

```
cols NMTOKEN #REQUIRED
```

There is no default. A value is required.

colsep= C.15

Applies to the CALS Table elements **Table**, **TGroup**, **ColSpec**, **SpanSpec**, **Entry**, and **EntryTbl**. A value of 1 means that a vertical rule should separate columns, and a value of 0 means not. Values are inherited from higher-level elements. At the column or entry level, the attribute applies to the right-hand separator. The value for the last column is ignored.

```
colsep NMTOKEN #IMPLIED
```

No default is defined, meaning that it is left to the rendering application to decide.

<ColSpec> C.16

Specifications for a table column. There should be one for each column in a **TGroup** (T.14). If two or more columns are spanned by a cell in any row, the columns should be named with **colname** (C.12), and a **SpanSpec** (S.20) will refer by name to the first and last columns spanned.

```
<!ELEMENT ColSpec - O EMPTY>
<!ATTLIST ColSpec
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          align (left|right|center|justify|char) #IMPLIED
          char CDATA #IMPLIED
          charoff NMTOKEN #IMPLIED
          colname NMTOKEN #IMPLIED
          colnum NMTOKEN #IMPLIED
          colsep NMTOKEN #IMPLIED
          colwidth CDATA #IMPLIED
          rowsep NMTOKEN #IMPLIED
          TIM2 NMTOKEN #IMPLIED>
```

columns= C.17

Number of columns used when the arrangement of a **SimpleList** (S.15) is specified to be horizontal or vertical (e.g., **<SimpleList arrange="horiz" columns=3>**).

```
columns NMTOKEN #IMPLIED
```

There is no default, meaning that it is left to the rendering application to determine. A value of "1" makes the simple list look like an ordinary block list.

colwidth= C.18

Applies to the CALS Table element **ColSpec**. Specifies the width of a column. It may be an absolute measurement, e.g., "3pc" for 3 picas, "36pt" for 36 points;

standard measurement units and abbreviations are not specified, but *pt*, *cm*, and *in* may be adequate for most purposes; fractional units must be expressed as "0.125in". Or the value may be relative, "1*" or "*" for one unit, "2*" for two units, twice as wide as 1*).

```
colwidth CDATA #IMPLIED
```

Units of measure that should be understood: "mm" (millimeter); "cm" (centimeter); "in" (inch); "p" (pica, 1/6 in), "pt" (point, 1/72 in); "dd" (didot, 1/67.431 in); "cc" (cicero, 1/5.619 in, 12 didots); a number with neither specified units nor "*" should be interpreted as pixels. Use lower case. Two units of measure cannot be combined in one value (e.g., "3p6" should be "3.5p" or "42pt").

The default value is equivalent to "1*".

"Command", type= — T (T.1)

"Comment", type= — Annote (A.9)

%commonatts; C.19

An entity defining identifying attributes that are available for most elements but that have no practical use with "wrapper" elements like **Body** and **TitleGroup**, and so are not included in **%baseatts;**:

```
<!ENTITY % commonatts
         "emph (ital|bold|boldital|roman|und|dblund|color|sys) #IMPLIED
          alerts IDREFS #IMPLIED
          altreps IDREFS #IMPLIED">
```

Whenever "%commonatts;" appears in an element's ATTLIST, the three attribute declarations above are substituted for it. Refer to the individual entries for the meanings and uses of each: **emph** (E.6), **alerts** (A.3), and **altreps** (A.6).

<Company> C.20

The **Company** element contains the common name of the document's owner:

```
<Company>Alcatel</Company>
```

The exact phrasing is left to the document provider. It need not be the legal name nor the shortest commonly used name, but it should be specific enough to be unique within the industry and recognizable to most people who work in it. The same name should be used for all of the company's documents. It should generally not be the company's COMMON LANGUAGE code, which is part of the **EDocID** (E.1), or anything else as cryptic as that.

```
<!ELEMENT Company    - -  (#PCDATA)* >
<!ATTLIST Company
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
```

```
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

This is a required subelement of the **DocID** (D.27).

"ConstantWidth", type= — T (T.1)

<Contact> C.21

May occur any number of times within **DocID** (D.27) to provide information about the persons or entities to be contacted for more information on the subject matter of the document.

```
<!ELEMENT Contact    - -  (#PCDATA)* >
<!ATTLIST Contact
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Multiple **Contact** elements should be used for multiple contacts. Specific uses of this element and specific forms for the information it contains are left to the discretion of the document provider. Examples:

```
<Contact>Sheena Ng, Bellcore, 445 South St., Morristown, NJ,
09760, (201) 829-0000</Contact>

<Contact>ITGR Bulletin Board: (201) 829-0000 (9600): (201) 829-
9999 (14.4)</Contact>
```

<Contents> C.22

A list of contents, or a specification for one. May appear within any of the major subdivisions of **TDoc** or any level of **Section**, and may be a typical Table of Contents (listing lower-level section titles and page numbers), a List of Figures, a List of Tables, or something atypical. It is not required to appear at the beginning of a section. It may (should?) contain hypertext links rather than page numbers. Its content model and attributes are exactly the same as those of **Index** (I.9) — the difference is entirely in the ordering of the list: A contents list has items in the order they appear in the document, while an index lists items in alphanumeric order.

To specify how the contents list is to be constructed, special attributes are available:

elements: elements to be included in the list ("Section", "Equation", "Frame", etc.).

levels: nested levels of the elements to be included (e.g., "1 2 3" for the first 3 levels).

groups: values of the **group** attribute to be included in the list. That is, only elements that have one of the listed values of the **group** attribute should be included in the list.

types: **type** and/or **dstype** values to be included in the list.

remaps: **remap** values to be included in the list.

langs: languages to be included in the list.

revstats: revision statuses to be included in the list.

statuses: values of **status** to be included in the list.

scope: where to look for elements to list or index: All elements listed should be searched for subelements matching the preceding criteria. The default value is undefined, meaning that the whole document is to be searched. ("Body" may be a more useful value for most purposes.) The special value "Parent" means the parent element of the current **Contents** or **Index** element, for section-level contents lists and indexes. **Contents** and **Index** elements themselves should not be searched. (This does not mean that an Index cannot be listed in a Table of Contents — it will be if the value for **elements** includes ". . . Index . . .". It does mean that subelements of contents lists and indexes cannot be included.)

include: what information about each element should be included in the list. Values that should be recognized: **num** (element number or mark), **title** (element title), **text** (element text), **allcontent** (element content of all forms), **page** (page number), **link** (hypertext link from contents list item to element).

When more than one value is provided for one of these attributes, the values are logically OR'd (i.e., they expand the list): for instance, **types="Figure Exhibit"** means list elements with either "Figure" or "Exhibit" as the value of **type**. Values of separate attributes are logically AND'd (i.e., they restrict the list): for instance, **elements="T" types="DocID"** means list only those elements that are text phrases *and* have the value "DocID" for the **type** attribute (presumably that would not be all the text phrases).

A **Contents** element must have an endtag ("**</Contents>**"), even if it has only a specification and no supplied content.

```
<!ELEMENT Contents - - ((Title|TitleGroup)?,(P|S|Annote|Frame|Danger
          |Caution|Warning|Admon|Table|OrderedList|UnorderedList
          |VariableList|SegmentedList|Listing|Pt|IndexEntry|Graphic
          |Object|ExternalText|Flowchart|Contents)*) >
<!ATTLIST Contents
          %baseatts;
          %identity;
          elements NMTOKENS #IMPLIED
          levels NMTOKENS #IMPLIED
          groups NMTOKENS #IMPLIED
          types NMTOKENS #IMPLIED
          remaps CDATA #IMPLIED
          langs CDATA #IMPLIED
```

```
revstats NMTOKENS #IMPLIED
statuses CDATA #IMPLIED
scope NMTOKENS #IMPLIED
include NMTOKENS #IMPLIED >
```

Values of the **type** attribute that should be recognized: type values to expect: "Contents" (a typical TOC); "Figures"; "Exhibits"; "Tables"; "Equations"; "Procedures"; "RqmtObjs" (requirement objects); "Attachments"; "Revisions". See the entries for the entities **%baseatts;** (B.2) and **%identity;** (I.3) for more on this element's attributes.

Comment

An author who does not wish to make assumptions about the capabilities of the rendering application will include both specifications (in the form of attribute values) and content for a **Contents** element. Typically, an unsophisticated rendering application would present the supplied content, while a more capable one could replace that with generated content matching the specifications. However, rendering applications are free to ignore both the supplied content and the specifications, and even the presence or absence of a **Contents** element, for the sake of consistency of presentation across a document set. Authors should particularly consider the futility of supplying page numbers when the format and medium of presentation are indeterminate. (For almost all other elements, only formatting instructions, like numbering, may be ignored; content may not be.)

Part of the consideration for authors must be that there is as yet no application that can implement all the features of TIM contents-list generation.

contents= (TIM 1 only) — present (P.16)

"Contents", type= — Contents (C.22)

continuation= C.23

Specifies whether the number of a section, list item, frame, paragraph, etc., should be one more than the number of the previous such element, or should restart at 1 (or the number specified by the **restartsat** attribute (R.15).

```
continuation (continues|restarts|holds) #IMPLIED
```

The **continuation** attribute is defined, for the elements that have it, through the entity reference **%numbering;** (N.13).

The implied default is for numbering of block lists (**OrderedList**, **SegmentedList**, and **VariableList**) to restart at 1 with each new list but continue for the second and further parts of the same list (identified by **OrdListGrp**, etc.). The attribute applies to individual section, para, frame, and distributed-list-item elements, so the default for these is "restarts" for the first in each number stream, "continues" for the rest. The value "holds" is used with

split numbering, to indicate that the value of the main number remains the same: 1 (restarts), 2a (continues), 2b; or 1 (restarts), 1a (holds), 1b.

Note the general rule that the value of **`continuation`** directly affects only the first item in a list or list group. The remaining items increment by one.

Examples (as the implied default, **`continuation="restarts"`** can be omitted for the first **`OrdListGrp`**):

If the document contains:	**Then the likely markup is:**
1. First item 2. Second item 4. Next item, there is no item 3	<OrderedList> <OrdListGrp numeration=”arabic” suffix=”.”> <LI label=”1.”><P>First item</P></LI> <LI label=”2.”><P>Second item</P></LI></OrdListGrp> <OrdListGrp numeration=”arabic” continuation=”restarts” restartsat=”4” suffix=”.”> <LI label=”4.”><P>Next item, there is no item 3</P></LI></OrdListGrp> </OrderedList>
1. First item 2a. Second item, part a 2b. Second item, part b	<OrderedList> <OrdListGrp numeration=”arabic” suffix=”.”> <LI label=”1.”><P>First item</P></LI></OrdListGrp> <OrdListGrp numeration=”arabic” splitnum=”splitlalpha” suffix=”.”> <LI label=”2a.”><P>Second item, part a</P></LI> <LI label=”2b.”><P>Second item, part b</P></LI></OrdListGrp> </OrderedList>
1. First item 1a. First item, subpart a 1b. First item, subpart b	<OrderedList> <OrdListGrp numeration=”arabic” suffix=”.”> <LI label=”1.”><P>First item</P></LI></OrdListGrp> <OrdListGrp numeration=”arabic” splitnum=”splitlalpha” continuation=”holds” suffix=”.”> <LI label=”1a.”><P>First item, subpart a</P></LI> <LI label=”1b.”><P>First item, subpart b</P></LI></OrdListGrp> </OrderedList>

"continues", continuation= (C.23)

<Copyrt> C.24

May occur once within **DocID** (D.27) to provide copyright information about the document.

```
<!ELEMENT Copyrt    - -  (#PCDATA)* >
<!ATTLIST Copyrt
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Specific forms for the information it contains are left to the discretion of the document originator. Example:

```
<Copyrt>Copyright &copy; 1993, 1995 Bellcore. All Rights
Reserved.</Copyrt>
```

"Copyrt", type= — T (T.1)

<Country> C.25

An optional element of **DocID** (D.27) used to identify the country or countries for which this version of the document is produced.

```
<!ELEMENT Country    - -  (#PCDATA)* >
<!ATTLIST Country
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Content and format are left to the document provider:

```
<Country>UK</Country>

<Country>England, Scotland, Wales, Northern Ireland</Country>
```

If a country code is used to distinguish separately orderable versions of document, that code should appear in a **DN.country** (D.12) subelement of **DocNo**, whether or not it also appears in **Country**.

CURRENT (SGML reserved name, TIM 1 only) C.26

Reserved name that may be used in place of a default value for an attribute. This allows an attribute value to be "inherited" by default from a previous element of the same type. **Locsrc** (L.15) was defined this way in TIM 1, consistent with the HyTime standard, but CURRENT is not permitted in XML (X.2), so the value of **locsrc** is IMPLIED (I.6) in TIM 2.

D (Danger — DXF)

<Danger> D.1

An admonishment that warns of the risk of serious or fatal injury or illness. Admonishments — the other specific types are **Admon** (A.2), **Caution** (C.3) and **Warning** (W.1) — are important in telecom documents, and ideally they should be given sophisticated presentation: they generally apply to a span of text, and should appear automatically whenever any part of that text is presented (see the comment and example at **Caution**). They have the data-content model of an **S** structure (S.1).

```
<!ELEMENT Danger - - (Title?,(P|S|Annote|Frame|Table|OrderedList
          |UnorderedList|VariableList|SegmentedList|Listing|Pt|IndexTerm
          |Graphic|Object|Flowchart|ExternalText)*) >
<!ATTLIST Danger
          %baseatts;
          %identity;
          %numbering;
          %commonatts;>
```

See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), and **%commonatts;** (C.19) for all this element's attributes. Use of the **type** attribute is not supported.

It is expected that the meanings, uses, and presentation specifications of admonishments will be the subject of a separate TCIF Guideline.

<DataLoc> D.2

HyTime element that creates links to specific streams of data. The start and end points of the linked-to data stream are specified by pairs of numbers that count off the kinds of units specified by the **quantum** attribute (Q.1).

```
<!ELEMENT DataLoc - - (#PCDATA)>
<!ATTLIST DataLoc
          id ID #REQUIRED
          HyTime NMTOKEN #FIXED "DataLoc"
          locsrc IDREFS #IMPLIED
          quantum (str|norm|word|name|sint|date|time|utc) "norm"
          catsrc (catsrc|catsrcsp|nocatsrc) "nocatsrc"
          catres (catres|catressp|nocatres) "nocatres">
```

The **locsrc** attribute references the ID of the element that contains the data string; for external links, **locsrc** will point to either a **NameLoc** (N.3) or **TreeLoc** (T.22) element. **catsrc** (C.2) and **catres** (C.1) are meaningful only for links to multiple targets.

Example for an internal cross-reference:

```
<Ref rid="TCIF-IPI-95-004-XREF-1">...</Ref><DataLoc id="TCIF-
IPI-95-004-XREF-1" locsrc="TCIF-IPI-95-004-LISTING-908">10 11</
DataLoc>
```

This would link to (or retrieve) the 10th through 20th words (i.e., start at word 10, count 11 words) in the element with the ID "TCIF-IPI-95-004-LISTING-908".

Example for an external cross-reference (the external document must be given an entity name in the current document's prologue):

```
<!DOCTYPE TDoc PUBLIC "-//USA/TCIF//DTD TIM-2//EN" [

...

<!ENTITY Tedd SYSTEM "../../../ipi95001/mtext/main.tim" NDATA
SGML>

...

]>

<TDoc>

...

...<Ref rid="TCIF-IPI-95-004-XREF-2"> (see "The TEDD
Package")</Ref><DataLoc id="TCIF-IPI-95-004-XREF-2"
locsrc="TCIF-IPI-95-004-NAMELOC-1">1 1</DataLoc><NameLoc
id="TCIF-IPI-95-004-NAMELOC-1"><NmList docorsub="Tedd">TCIF-
IPI-95-001-XREF-342</NmList></NameLoc>
```

The content of **`NmList`** is the ID of the target element in the external document. The **`DataLoc`** now specifies only the first word in the element.

"Date", type= — T (T.1)

"date", quantum= (Q.1)

dbid= (TIM 1 only) — rid (R.18)

dbids= (TIM 1 only) — rid (R.18)

<DBLink> (TIM 1 only) — Link (L.9)

"dblund", emph= (E.6)

"Decision", type= — LI (L.8)

"Definition", type= — LI (L.8), T (T.1)

"delete", type= — revstat (R.16)

dimensions= D.3

Preferred horizontal and vertical dimensions of a **`Frame`**, **`Graphic`**, or presentation window for an **`Object`**.

```
dimensions NMTOKENS #IMPLIED
```

Examples:

```
<Graphic dimensions="640 480" ...
<Frame dimensions="8.5in 11in" ...
```

Units of measure that should be understood: pixels (default, no abbreviation); "mm" (millimeter); "cm" (centimeter); "in" (inch); "p" (pica, 1/6 in), "pt" (point, 1/72 in); "dd" (didot, 1/67.431 in); "cc" (cicero, 1/5.619 in, 12 didots). Use lower case. Two units of measure cannot be combined in one value (e.g., "3p6" should be "3.5p" or "42pt").

<DimList> (TIM 1 only) D.4

A HyTime element in TIM 1 that contained the **DimSpec**s, dimension specifications, for a **DataLoc**. Although its content is always supposed to be number pairs (start and end points), the content model doesn't require explicit tagging of each pair (all enclosed numbers would be read off in pairs regardless of any **DimSpec** or **MarkList** tags).

<DimSpec> (TIM 1 only) D.5

A HyTime element in TIM 1 that contained marker pairs that delineated start and end points of a link end specified by a **DataLoc**. Its explicit use is never required for HyTime compatibility, so it has been dropped from TIM 2.

"Disclaimer", type= — Section (S.4)

<DistOrdLI> D.6

An item of a distributed ordered list, an ordered list in which the items are not contiguous (and therefore cannot be contained in a single parent element like **OrderedList**, O.3). **DistOrdLI** has the same content model as an **LI** element (L.8), meaning that it can contain one paragraph or many paragraphs and sublists. But it cannot contain another distributed-list item. Each **DistOrdLI** should have a **type** attribute identifying the list to which it belongs, since it may be intermingled with **DistOrdLI**'s from other lists.

If you want titles (column heads) over the label and body of the **DistOrdLI**, you must repeat them for each **DistOrdLI** (in a block list, they are specified only in the parent **OrderedList** or **OrdListGrp**, not in the **...LI**s). If the item has an overall **Title**, the item's mark (number) will precede the title rather than the body of the item.

```
<!ELEMENT DistOrdLI - - (Title?,ListHead?,(P|S|Annote|Frame|Danger
          |Caution|Warning|Admon|Table|OrderedList|UnorderedList
          |VariableList|SegmentedList|Listing|Pt|IndexTerm|Graphic
          |Object|Flowchart|ExternalText)*) >
<!ATTLIST DistOrdLI
          %baseatts;
```

```
%identity;
%numbering;
%commonatts; >
```

Values of the **type** attribute that should be recognized: "Equation", "Example", "Issue", "Procedure," "RqmtObj" (requirement object), "Rule", "Step", and "TableList" (**DistOrdLI**'s used across cells of a table for what would otherwise be an **OrderedList**). (See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), and **%commonatts;** (C.19) for more on this element's attributes.

Comment

Distributed lists must be used carefully. Parsers will not catch most errors in using them: They will not care whether all items of a given list have the same value of **type** (they should); whether that value is different from the values for other distributed lists (it should be); or whether the numbering style has changed from item to item (if it does, how will readers know the various non-consecutive items constitute a list?).

Comment

There is no DistUnordLI. **DistOrdLI** can have numeration set to "none" and a label supplying a bullet or other mark, to make it what a DistUnordLI would be. This is not encouraged, because the relatedness of the list items could easily be overlooked without consecutive numbers.

Comment

Distributed lists do not exist in the DocBook DTD or any other as far as is known. But equations, requirements, and procedures are often rendered using such constructs.

A particular use of **DistOrdLI** is in numbering across cells of a table. An ordinary list could not span cells because each cell element (**Entry**) is the parent of the cell contents, so block lists have to be entirely in one cell. (Tables with numbered rows are better represented in TIM as **SegmentedLists**, S.7.)

Comment

All ordered lists could be marked up using distributed list items, but only by imposing a requirement that a **type** be defined for every list item. Since that is not normal practice and since many authors will never need distributed lists, **OrderedList** and **DistOrdLI** coexist in TIM.

<DistSegLI> (TIM 1 only) —

Dropped from TIM 2 as unnecessary. See **SegmentedList** (S.7).

<DistVarLI> D.7

A list item of a distributed variable list — which is like a **VariableList** (V.2) except that the items do not appear contiguously. Each **DistVarLI** should have a **type** attribute identifying the list to which it belongs, since it is separated from other items in the list.

If you want titles (i.e., column heads) over the **ListTerm** and the **LI** that make up the **DistVarLI**, you must repeat them for each **DistVarLI** (in a block list, they are specified only in the parent **VariableList** or **VarListGrp** item, not in the **VarLI**s). A title cannot be placed over the item number, which is rarely used. If the **DistVarLI** has a title, the item's mark (number) will precede it rather than the body of the item.

A **DistVarLI** may contain multiple **ListTerm**s or **LI**s; this is allowed but it may not be well supported by applications.

```
<!ELEMENT DistVarLI - - (Title?,ListHead?,ListTerm+,Pt*,LI,(Pt|LI)*) >
<!ATTLIST DistVarLI
          %baseatts;
          %identity;
          %numbering;
          %commonatts; >
```

Values of the **type** attribute that should be recognized: "Message" and "RqmtObj" (requirement object). Other types should be indicated by the **type** value of the enclosed **ListTerm** (L.14): "Acronym", "GlossTerm", etc. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), and **%commonatts;** (C.19) for more on this element's attributes.

<DN.addm> D.8

Addendum number, an optional subelement of a formal document identifier, **DocNo** (D.28).

```
<!ELEMENT DN.addm    - -  (#PCDATA)* >
<!ATTLIST DN.addm
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

<DN.apdx> D.9

Appendix number, an optional subelement of a formal document identifier, **DocNo** (D.28).

```
<!ELEMENT DN.apdx    - -  (#PCDATA)* >
<!ATTLIST DN.apdx
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
```

```
            remapatt CDATA #IMPLIED
            remapto CDATA #IMPLIED
            TIM2 NMTOKEN #IMPLIED >
```

<DN.base> D.10

Base number, the required first subelement of a formal document identifier, **DocNo** (D.28).

```
<!ELEMENT DN.base    - -  (#PCDATA)* >
<!ATTLIST DN.base
            id ID #IMPLIED
            remap NMTOKEN #IMPLIED
            remapatt CDATA #IMPLIED
            remapto CDATA #IMPLIED
            TIM2 NMTOKEN #IMPLIED >
```

<DN.bull> D.11

Bulletin number, an optional subelement of a formal document identifier, **DocNo** (D.28).

```
<!ELEMENT DN.bull    - -  (#PCDATA)* >
<!ATTLIST DN.bull
            id ID #IMPLIED
            remap NMTOKEN #IMPLIED
            remapatt CDATA #IMPLIED
            remapto CDATA #IMPLIED
            TIM2 NMTOKEN #IMPLIED >
```

<DN.country> D.12

Country, an optional subelement of a formal document identifier, **DocNo** (D.28). Use this element only if country is used (say, in a field of a catalog database) to identify different versions of the document. If something other than a country code distinguishes the document for ordering purposes but you want to specify the country, use the **Country** element (see comment at **DocNo**).

```
<!ELEMENT DN.country    - -  (#PCDATA)* >
<!ATTLIST DN.country
            id ID #IMPLIED
            remap NMTOKEN #IMPLIED
            remapatt CDATA #IMPLIED
            remapto CDATA #IMPLIED
            TIM2 NMTOKEN #IMPLIED >
```

<DN.customer> D.13

Customer, an optional subelement of a formal document identifier, **DocNo** (D.28). Use this element only if customer is used (say, in a field of a catalog

database) to identify different versions of the document (see comment at **DocNo**).

```
<!ELEMENT DN.customer    - -  (#PCDATA)* >
<!ATTLIST DN.customer
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

<DN.date> D.14

Date, an optional subelement of a formal document identifier, **DocNo** (D.28). Use this element only if date is used rather than an issue, edition, or revision number to identify different versions of the document. The document's date must appear in the **DocDate** element (D.25).

```
<!ELEMENT DN.date    - -  (#PCDATA)* >
<!ATTLIST DN.date
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

<DN.ed> D.15

Edition number, an optional subelement of a formal document identifier, **DocNo** (D.28).

```
<!ELEMENT DN.ed    - -  (#PCDATA)* >
<!ATTLIST DN.ed
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

<DN.iss> D.16

Issue number, an optional subelement of a formal document identifier, **DocNo** (D.28).

```
<!ELEMENT DN.iss    - -  (#PCDATA)* >
<!ATTLIST DN.iss
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

<DN.lang> D.17

Language, an optional subelement of a formal document identifier, **DocNo** (D.28). Use this element only if language is used (say, in a field of a catalog database) to identify different versions of the document. If something other than a language code distinguishes the document for ordering purposes but you want to specify the language, use the **Lang** element (L.2). See comment at **DocNo**.

```
<!ELEMENT DN.lang    - -  (#PCDATA)* >
<!ATTLIST DN.lang
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

<DN.prod> D.18

Product (the product the document is relevant to), an optional subelement of a formal document identifier, **DocNo** (D.28). Use this element only if product is used (say, in a field of a catalog database) to identify different versions of the document. If something other than a product code distinguishes the document for ordering purposes but you want to specify the product, use the **ProdDsc** element (P.19). See comment at **DocNo**.

```
<!ELEMENT DN.prod    - -  (#PCDATA)* >
<!ATTLIST DN.prod
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

<DN.rev> D.19

Revision number, an optional subelement of a formal document identifier, **DocNo** (D.28). In current common practice, use of a revision number rather than a revision-level number indicates that the information product is a revision package rather than a complete document incorporating the revisions. Complete documents are expected under current TCIF guidelines, implying that this element should not be used until further notice.

```
<!ELEMENT DN.rev    - -  (#PCDATA)* >
<!ATTLIST DN.rev
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

<DN.rlse> D.20

Release number, an optional subelement of a formal document identifier, **DocNo** (D.28).

```
<!ELEMENT DN.rlse    - -  (#PCDATA)* >
<!ATTLIST DN.rlse
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

<DN.rvl> D.21

Revision-level number, an optional subelement of a formal document identifier, **DocNo** (D.28). In current common practice, use of a revision-level number rather than a revision number indicates that the information product is a complete document incorporating revisions, rather than a separately cataloged, separately orderable revision package.

```
<!ELEMENT DN.rvl    - -  (#PCDATA)* >
<!ATTLIST DN.rvl
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

<DN.supp> D.22

Supplement number, an optional subelement of a formal document identifier, **DocNo** (D.28).

```
<!ELEMENT DN.supp    - -  (#PCDATA)* >
<!ATTLIST DN.supp
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

<DN.vol> D.23

Volume number, an optional subelement of a formal document identifier, **DocNo** (D.28).

```
<!ELEMENT DN.vol    - -  (#PCDATA)* >
<!ATTLIST DN.vol
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
```

```
            remapto CDATA #IMPLIED
            TIM2 NMTOKEN #IMPLIED >
```

DOC (notation) D.24

Notation that should be declared for a `filename.doc` Microsoft Word file.

```
<!NOTATION DOC       SYSTEM "Microsoft Word document">
```

Example:

```
<!ENTITY doc001 SYSTEM "../other/doc001.doc" NDATA DOC>

...

<Object entityref="doc001">
```

The file should be stored in an `other` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s). DOC files may be included in TEDD packages experimentally. Preferred formats will be specified in future TCIF Guidelines.

<DocDate> D.25

The document's publication date, a required subelement of **DocID** (D.27).

```
<!ELEMENT DocDate    - -  (#PCDATA)* >
<!ATTLIST DocDate
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

The date should be expressed in year-month-day format, with the day optional. The year may be expressed in either two-digit format (e.g., 95) or four-digit format (e.g., 1995); month and day should each be two digits. The solidus or the hyphen may be used to separate date fields (e.g., 95/08/01 or 95-08-01).

Preferred form:

```
<DocDate>1993-08-21</DocDate>
```

Other acceptable forms:

```
<DocDate>93-08-21</DocDate>

<DocDate>93/08/21</DocDate>

<DocDate>93-08</DocDate>

<DocDate>1993/08</DocDate>
```

<DocDsc> D.26

May occur once within the **DocID** element (D.27) to provide information for which no other element of DocID is appropriate.

```
<!ELEMENT DocDsc    - -  (#PCDATA)* >
<!ATTLIST DocDsc
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Content and format are left to the document provider. All content will be treated as a single paragraph; no special formatting will be recognized.

Example:

```
<DocDsc>Document Type: Product Functional Specification</
DocDsc>
```

<DocID> D.27

The container for the document's required and optional identifying information: originating company, document number, issue number, title, date, copyright, keywords, etc.

```
<!ELEMENT DocID - - (EdocID,Class,Company,(DocNo|DocNo.u),DocDate,
          TitleGroup,VersionControl?,AltNo*,Publisher?,ISN*,Country*,
          Copyrt?,Lang*,(SuperDoc|SubDoc)*,CDocType?,MTextType,
          DraftStatus?,Status*,PropStat?,PropMsg?,EdocRepl*,DocRepl*,
          ProdDsc*,DocDsc?,RelDoc*,OrderInfo?,Contact*,Au*,Abs?,
          Keywords?)>
<!ATTLIST DocID
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Seven subelements are required: **EdocID** (E.1), **Class** (C.10), **Company** (C.20), **DocNo** (D.28, or **DocNo.u**, D.29), **DocDate** (D.25), **TitleGroup** (T.21), and **MTextType** (M.9).

"DocID", type= — T (T.1)

<DocNo> D.28

The **DocNo** element contains a formally structured document number. It can be replaced by the unstructured **DocNo.u** element (D.29), but **DocNo** should be used whenever possible.

```
<!ELEMENT DocNo - - (DN.base,(DN.iss|DN.ed|DN.addm|DN.rev|DN.rvl|DN.vol
```

```
          |DN.apdx|DN.bull|DN.supp|DN.rlse|DN.lang|DN.date|DN.prod
          |DN.country|DN.customer)*)>
<!ATTLIST DocNo
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Example:

```
<DocNo><DN.base>GR-1383-CORE</DN.base><DN.iss>1</DN.iss></
DocNo>
```

DocNo content consists of subelements whose names begin with "DN": **DN.base** (D.10) is required; the others are optional and may occur in any order, to allow specification of a document identifier in any company's style.

Comment

It is assumed that any and all — and only — information the document owner uses to uniquely identify the particular information product will be included within appropriate "DN" subelements. Information that is merely descriptive should not appear within **DocNo**: thus there are both **DN.date** and a separate **DocDate**, **DN.lang** and a separate **Lang** element, etc. The "DN" elements should be used only when the information is part of the unique identifier — that is, when it may be necessary for ordering or locating the right document. Issue, edition, or revision number, and supplement, volume, or appendix number, if any, are normally part of the unique identifier. Date normally is not, unless it is used in place of an issue or revision number. Language, customer, country, or product would often be needed to distinguish different versions of a document, but they may also be encoded into the base number, in which case use of a separate subelement would be redundant.

<DocNo.u> D.29

The **DocNo.u** element contains an unstructured document number. It can be used in place of the structured **DocNo** element when the subelements of **DocNo** aren't adequate to represent the full identifier of the document.

```
<!ELEMENT DocNo.u    - -  (#PCDATA)* >
<!ATTLIST DocNo.u
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Either **DocNo** or **DocNo.u** must be present within **DocID** (D.27). **DocNo** is recommended. Examples of **DocNo.u**:

```
<DocNo.u>XRV-331 ISS 1 PROD 1</DocNo.u>
```

```
<DocNo.u>XRV-331 ISS 1 PROD 2</DocNo.u>
<DocNo.u>XRV-331 ISS 1 PROD 2 RVL 1</DocNo.u>
```

docorsub= D.30

Applies to **NmList** (N.8). Specifies the name of the external entity that contains the external link target of a HyTime link. (The document prologue must contain an entity declaration with NDATA type of SGML for the target document.) See **NameLoc** (N.3) for an example.

```
docorsub ENTITY #IMPLIED
```

<DocRepl> D.31

Occurs any number of times within **DocID** to identify information products that have been replaced by the current one. Each instance contains a document number (**DocNo.u** or **DocNo**), and optionally a simple **Title** or complete **TitleGroup**. Compare **EdocRepl** (E.2).

```
<!ELEMENT DocRepl    - -  ((DocNo|DocNo.u),(Title|TitleGroup)?) >
<!ATTLIST DocRepl
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Example:

```
<DocRepl><DocNo><DN.base>SR-4392</DN.base><DN.iss>1
</DN.iss></DocNo><Title>Changes to Bellcore's Document
Identifiers</Title></DocRepl>
```

"DocTitle", type= — T (T.1)

"DocType", type= — SubTitle (S.27)

<DraftStatus> D.32

May occur once within **DocID** (D.27) to provide a simple statement of the draft status of the document (suitable as a message in the header, footer, or background of printed pages: "DRAFT", "DRAFT FOR REVIEW", "PRIVATE", etc.).

```
<!ELEMENT DraftStatus - - (#PCDATA)* >
<!ATTLIST DraftStatus
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Specific wordings are left to the discretion of the document originator.

dstype= D.33

Domain-specific type. Indicates that the element in question contains a domain-specific structure (one that would be recognized by only some kinds of applications). This attribute is usually used to mark highly structured objects, such as literal program code or database fields, and the **dstype** value would normally be meaningful only to a compiler, a database system, or some other specialized application. **Dstype** is not for local extensions to **type** (T.23); those should simply be locally agreed-upon values for **type**. The reason for the distinction is that an element with a **dstype** attribute is likely to have an **type** attribute as well, meaningful to nonspecialized applications like browsers and publishing systems: e.g., **type="Code" dstype="C++"**. On the other hand, when local practice is to have specific subcategories of **type**, these are better conveyed in **dstype**, so that recipients can use either the general and familiar **type** or the more specific **dstype** value: e.g., **type="RqmtObj" dstype="ConditionalRequirement"**.

```
dstype CDATA #IMPLIED
```

The default value of **dstype** is undefined.

This attribute is defined for all elements that have the **%baseatts;** (B.2).

dtdorlpd= D.34

Applies to **NmList** (N.8). Specifies an alternative DTD or Link Process Definition (LPD) to be used to parse the target document. To use this attribute, you must modify the TCIF-standard SGML declaration (in the file `tcif.dcl`) to turn on the CONCUR and/or LINK features.

```
dtdorlpd NMTOKENS #IMPLIED
```

DXF (notation) D.35

Notation that should be declared for a `filename.dxf` AutoCad Drawing Exchange Format file.

```
<!NOTATION DXF       SYSTEM "DXF">
```

Example:

```
<!ENTITY fig001 SYSTEM "../dxf/fig001.dxf" NDATA DXF>
```

The file should be stored in a `dxf` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s). DXF files may be included in TEDD packages experimentally. Preferred formats will be specified in a future TCIF Guideline. Currently the options for vector-graphic files and metafiles are CGM (C.6), CGM-CLEAR (C.7), DXF (D.35), EPS (E.12), PDF (P.5), PICT (P.9), WMF (W.5), and WPG (W.6).

E (EdocID — extra)

"E-Address", type= — (T.1)

"Editor", type= — (T.1)

"EditorNote", type= — Annote (A.9)

<EdocID> E.1

The unique ID for the particular electronic instance of a **TDoc** document; it must be different for each different instance.

```
<!ELEMENT EdocID    - -  (#PCDATA)* >
<!ATTLIST EdocID
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

In telecom-industry use, the content must begin with the unique code for the owner of the document, followed by a hyphen. After that, there should be an identifier unique among all documents delivered electronically by that owner. It is recommended, not required, that some form of the document's official number be included, but there must also be enough more to uniquely identify each distinct instance of that document — that is, if an error is corrected in a constituent file and the document is delivered a second time, the **EdocID** should be changed. More advice is given in the TCIF Guideline document TCIF-IPI-95-001, *The Telecommunications Electronic Document Delivery (TEDD) Package*.

Examples:

```
<EdocID>TCIF-IPI-95-004:issueA:instance1</EdocID>

<EdocID>BR-GR-1383-CORE:issue1:instance2</EdocID>
```

<EdocRepl> E.2

Occurs any number of times within **DocID** to identify previously delivered TEDD-packaged document instances replaced by this instance.

```
<!ELEMENT EdocRepl    - -  (#PCDATA)* >
<!ATTLIST EdocRepl
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Example:

```
<EdocID>BR-GR-1383-CORE:issue1:instance2</EdocID>
```

```
...
<EdocRepl>BR-GR-1383-CORE:issue1:instance1</EdocRepl>
```

In this example, there would not be a corresponding **DocRepl** (D.31), since what is replaced is an earlier electronic instance of the same document. Conversely, there could be a **DocRepl** with no corresponding **EdocRepl** if the replaced document was never delivered in a TEDD package.

ELEMENT (SGML Reserved Name) E.3

Introduces the element declaration for every SGML element. It always has these parts:

1. The element name
2. The start- and end-tag omission rules: "- -" if neither can be omitted, "- O" if the end-tag can be omitted, "O O" if both can be omitted
3. The content model (in parens), which says
   - what subelements and/or data types can occur in the element
   - in what order they can occur: "," indicates sequence, "|" indicates any order
   - how many times they can occur: "?" means 0 or 1, "*" means 0 or more, "+" means 1 or more, no indicator means 1 time; "ElementX,ElementX+" can be used for 2 or more times

Two optional parts of an element declaration were used in TIM 1 but not in TIM 2: "+(...)" is a list of inclusions, additional elements that can appear anywhere in the element or any of its subelements (footnotes might be treated as inclusions), and "-(...)" is a list of exclusions, not allowed in the element or any of its subelements, even if their own declarations allow them. XML does not permit inclusions or exclusions (to avoid the considerable processing overhead), so TIM 1 inclusions were made explicit parts of each TIM 2 content model and TIM 1 exclusions (which were mostly to avoid bad consequences of inclusion rules) were dropped.

"element", nametype= (N.6)

elements= E.4

Specifies what elements should be listed in a **Contents** list or an **Index**. Typically, a **Contents** list would specify something like **elements="Section"** (for a Table of Contents) or **elements="Frame" types="Figure"** (for a List of Figures), and an Index would specify **elements="IndexTerm"**. See **Contents** (C.22) for an explanation.

```
elements NMTOKENS #IMPLIED
```

The default value of this attribute should be understood as "Section" for a **Contents** element and "IndexEntry" for an **Index** element (I.9).

"EMail", type= — T (T.1)

<Emph> E.5

Emphasis (**emph** is also an attribute, E.6). A text-level element with the same content model as a **T** phrase element. It is normal, but not strictly necessary, for a value of the **emph** attribute (E.6) to be specified within an **Emph** element — if it is not, the value "ital" is implied.

```
<!ELEMENT Emph - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef|IndexTerm
          |Pt|Graphic|Object|ExternalText|SimpleList|Equation
          |GraphicalText)*>
<!ATTLIST Emph
          %baseatts;
          id ID #IMPLIED
          emph (ital|bold|boldital|roman|und|dblund|color|sys) #IMPLIED
          %textatts; >
```

See the entries for the entities **%baseatts;** (B.2) and **&textatts;** (T.12) for more on this element's attributes. Use of the **type** attribute is not supported ("type" of emphasis is indicated by the value of the **emph** attribute).

emph= E.6

Type of emphasis (**Emph** is also an element, E.5). Indicates how an element is to be emphasized on presentation. The defined types of emphasis are "bold", "ital", "boldital", "roman", "und" (underline), "dblund", "color" and "sys" (system-specific, which might be blinking in an on-line system or small-caps on paper). A rendering application should be able to do *something* with each of these values to distinguish it from unemphasized text, but does not have to be able to do exactly what the value calls for. A very capable system will need rules to cover nested levels of emphasis — for instance, it is common typographic convention to toggle italics (<emph ital>*text*<emph ital>text ...) and cumulate underlining (<emph emph="und">text<emph emph="und">text ...).

```
emph (ital|bold|boldital|roman|und|dblund|color|sys) #IMPLIED
```

The default value is undefined (that is, rendering applications should assume that no value is assigned, no emphasis is intended) except for the **Emph** element, for which the implied default is "ital". The value "roman" is available to be used in place of "ital" when the text is already italicized, since it cannot be assumed that rendering applications will understand that they should indicate italics within italics by making text unitalic.

This attribute is defined for all elements that have the **%commonatts;** (C.19).

EMPTY (SGML Reserved Name) E.7

Reserved name for an empty content type. Elements that have a content model of EMPTY cannot contain any data characters or subelements, and they cannot have end tags. They exist, in general, to provide information to the rendering application about how the document is to be processed. Some empty elements (including **Graphic**, **ExternalText** and **Object** in TIM) are placeholders for external entities; they are replaced with the appropriate entity (a figure, equation, etc.) by the rendering application. Some empty elements (such as **Link**) provide the rendering application with specifications for establishing hyperlinks between elements. TIM's general-purpose empty element is **Pt** (P.23).

ENTITY (SGML Reserved Name) E.8

Reserved name for the value type of an attribute that references an entity. Attributes that have a value type of ENTITY (such as **docorsub** or **entityref**) are used to create links to external entities, such as graphics files or other documents. When using such an attribute, the entity name should be given without the markup characters that surround entities in element content:

```
<Graphic entityref="fig001">
```

Not "&fig001;".

"entity", nametype= (N.6)

entityref= E.9

Entity reference. Indicates that an **ExternalText**, **Graphic**, or **Object** element's contents are contained in the entity identified by this attribute.

```
entityref ENTITY #REQUIRED
```

When using this attribute, the entity name should be given without the markup characters "&" and ";" that surround entities in element content:

```
<!ENTITY fig001 SYSTEM "../gif/fig001.gif" NDATA GIF>

...

<Graphic entityref="fig001">
```

<Entry> E.10

The content of a table cell. There should be one for each column in each row, unless a **SpanSpec** (S.20) indicates otherwise. An **EntryTbl** (E.11) may be substituted for an **Entry**.

```
<!ELEMENT Entry - - (P|S|Annote|Frame|Danger|Caution|Warning|Admon
          |DistOrdLI|DistVarLI|OrderedList|UnorderedList|VariableList
          |Listing|Pt|IndexTerm|Graphic|Object|ExternalText|Equation
```

```
          |GraphicalText)* >
<!ATTLIST Entry
          id ID #IMPLIED
          type NMTOKENS #IMPLIED
          dstype CDATA #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          group NMTOKENS #IMPLIED
          status CDATA #IMPLIED
          version IDREFS #IMPLIED
          revstat (norev|revised|revflag|insert|insflag|delete|strike)
            #IMPLIED
          lang CDATA #IMPLIED
          present (normal|float|alert|altrep) #IMPLIED
          presspec CDATA #IMPLIED
          align (left|right|center|justify|char) #IMPLIED
          char CDATA #IMPLIED
          charoff NMTOKEN #IMPLIED
          colname NMTOKEN #IMPLIED
          colsep NMTOKEN #IMPLIED
          morerows NMTOKEN "0"
          nameend NMTOKEN #IMPLIED
          namest NMTOKEN #IMPLIED
          rotate NMTOKEN "0"
          rowsep NMTOKEN #IMPLIED
          spanname NMTOKEN #IMPLIED
          valign (top|middle|bottom) #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Use of the **type** attribute is not supported.

<EntryTbl> E.11

A subtable within a table cell. May contain a sequence of structures each comparable to a **TGroup** (without **TFoot**). In TIM 1 and the original CALS TAble DTD, **EntryTbl** may not contain another complete **EntryTbl**, but the exclusion rule has been removed in TIM 2 for XML compatibility.

```
<!ELEMENT EntryTbl - - (ColSpec*,SpanSpec*,THead?,TBody) >
<!ATTLIST EntryTbl
          id ID #IMPLIED
          type NMTOKENS #IMPLIED
          dstype CDATA #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          group NMTOKENS #IMPLIED
          status CDATA #IMPLIED
          version IDREFS #IMPLIED
          revstat (norev|revised|revflag|insert|insflag|delete|strike)
            #IMPLIED
          lang CDATA #IMPLIED
          present (normal|float|alert|altrep) #IMPLIED
```

```
presspec CDATA #IMPLIED
align (left|right|center|justify|char) #IMPLIED
char CDATA #IMPLIED
charoff NMTOKEN #IMPLIED
colname NMTOKEN #IMPLIED
cols NMTOKEN #REQUIRED
colsep NMTOKEN #IMPLIED
nameend NMTOKEN #IMPLIED
namest NMTOKEN #IMPLIED
rowsep NMTOKEN #IMPLIED
spanname NMTOKEN #IMPLIED
tgroupstyle NMTOKEN #IMPLIED
TIM2 NMTOKEN IMPLIED >
```

Use of the **type** attribute is not supported.

Comment

Even without the nesting TIM 2 allows, **EntryTbl** is a processing nightmare to be avoided whenever possible. Spanning of the other rows in the same column and the other columns in the same row is an easier-to-implement alternative. That is, instead of splitting one table cell into multiple rows and columns, add that number of rows and columns to the whole table and span across the cells that don't need to be split.

EPS (notation) E.12

Notation that should be declared for a `filename.eps` Encapsulated PostScript file.

```
<!NOTATION EPS       PUBLIC "+//ISBN 0-201-18127-4::Adobe//NOTATION
          PostScript Language Ref. Manual//EN">
```

Example:

```
<!ENTITY fig001 SYSTEM "../eps/fig001.eps" NDATA EPS>
```

The file should be stored in an `eps` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s). EPS files may be included in TEDD packages experimentally. Preferred formats for vector-graphic files and metafiles will be specified in a future TCIF Guideline. Currently the options are CGM (C.6), CGM-CLEAR (C.7), DXF (D.35), EPS (E.12), PDF (P.5), PICT (P.9), WMF (W.5), and WPG (W.6). The TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG (P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18).

EQN (notation) E.13

A text notation containing **troff eqn** markup.

```
<!NOTATION EQN       SYSTEM "EQN">
```

It may used be within a TIM file:

```
<Equation format="EQN">.EQ ....
```

Or it may be in a separate file with a `.eqn` extension, in a `troff` directory/ folder at the same level — having the same parent — as the `mtext` directory/ folder containing the TIM file(s):

```
<!ENTITY eqn001 SYSTEM "../troff/eqn001.eqn" NDATA EQN>

...

<Equation><ExternalText entityref="eqn001"></Equation>
```

The EQN format may be included in TEDD packages experimentally. Preferred formats for equations will be specified in a future TCIF Guideline.

%eqns; E.14

An entity reference defined in TIM to allow new element declarations for equation markup to be included. This will be done if the document prologue contains an entity declaration for **`%eqns;`** that points to an actual file. Example:

```
<!ENTITY % eqns SYSTEM "../../lib/eqmarkup.dtd" NDATA SGML>
```

It is assumed that one of the declarations in the file would be for the element **`Equation`**, and the rest would be for subelements of **`Equation`**. Since TIM reads these declarations (if they exist) before its own declaration for **`Equation`**, the new ones will be in effect. By default, **`%eqns;`** is a null string, so its occurrence in the DTD does nothing:

```
<!ENTITY % eqns         "">
%eqns;
```

There are seven entity references for extensions to TIM 2, and their placement (before all element declarations in the DTD) and order (**`%other;`**, **`%memos;`**, **`%rqmts;`**, **`%procs;`**, **`%eqns;`**, **`%flows;`**, **`%tables;`**) are significant, since the order is the priority for redefining elements: only the first element and attribute declarations for a given element are used. The order shown reflects the likelihood that declarations in one extension would be redefining the elements in others: the table extension (already provided) is, of course, the least likely to affect others.

<Equation> E.15

An equation. It may occur in-line in running text or in a **`Frame`** (F.8), or conceivably elsewhere. It has the data-content model of a **`T`** text phrase (T.1). It is not numbered itself, but equation frames (**`<Frame type="Equation">`**) can be.

```
<!ELEMENT Equation - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef
          |IndexTerm|Pt|Graphic|Object|ExternalText|SimpleList|Equation
          |GraphicalText)* >
```

```
<!ATTLIST Equation
          %baseatts;
          %identity;
          %commonatts;
          %textatts;>
```

See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%commonatts;** (C.19), and **&textatts;** (T.12) for all this element's attributes. Use of the **type** attribute is not supported.

Comment

This release of TIM includes no markup for equations (that is, nothing comparable to **eqn** or **TeX** for marking up the components of an equation). Therefore an **Equation** element is likely to contain (or reference external entities containing) whatever forms of the equation are available from the encoding application:

```
<Frame type="Equation" numeration=arabic label="Equation
23."><Equation><Graphic entityref=eqn023GIF></Equation></Frame>
```

Or:

```
<Frame type="Equation" numeration=arabic label="Equation
23."><Equation altreps="BR-SR-3031-EQN23EQN BR-SR-3031-
EQN23GIF"><AltRep id="BR-SR-3031-EQN23EQN"><ExternalText
entityref=eqn023EQN></AltRep><AltRep id="BR-SR-3031-
EQN23GIF"><Graphic entityref=eqn023GIF></AltRep></Equation></
Frame>
```

Or:

```
<Frame type="Equation" numeration=arabic label="Equation
23."><Equation format="EQN" altreps="BR-SR-3031-EQN23GIF">[eqn
markup here]</Equation><AltRep id="BR-SR-3031-
EQN23GIF"><Graphic entityref=eqn023GIF></AltRep></Frame>
```

SGML markup for equations may be added to TIM 2 without editing the DTD by defining it in a separate file and using the %eqns; entity (E.14) as a pointer to that file in the document prologue:

```
<!ENTITY eqns SYSTEM "../../lib/eqmarkup.dtd" NDATA SGML>
```

"Equation", type= — DistOrdLI (D.6), Frame **(F.8)**

"Equations", type= — Contents (C.22)

"Equipment", type= — Admon (A.2)

"ErrorCode", type= — ListTerm (L.14)

"ErrorMessage", type= — T (T.1)

"ExecSum", type= — Section (S.4)

"Example", type= — DistOrdLI (D.6), Listing **(L.12)**

exclusions, in element declarations — ELEMENT (E.3)

"Exhibit", type= — Frame (F.8)

"Exhibits", type= — Contents (C.22)

<ExternalText> E.16

An external text entity (normally not SGML but another ASCII/ISO-646 notation). It may be either the entire content of an element (such as an **Equation**) or one of two or more alternative representations, in which case it should have its **present** attribute set to "altrep" and be referenced by another element's **altreps** attribute; the referencing element, a **Section**, **P**, **Table**, etc., should contain the preferred representation directly (that could also be in an **ExternalText** element.

An **ExternalText** element always references an external entity. This would normally be text with another kind of markup (a different DTD, or **troff tbl** or **eqn**, for instance). Generally, an alternative representation in TIM, such as text using a different natural language (e.g., French rather than English), should be in a "present= 'altrep'" element or in a marked section (M.2) within the file rather than in **ExternalText**, so that it can be parsed and processed as SGML markup (external text will bypass the parser).

```
<!ELEMENT ExternalText - O EMPTY >
<!ATTLIST ExternalText
          %baseatts;
          %identity;
          %commonatts;
          entityref ENTITY #REQUIRED >
```

Use of the **type** attribute is not supported. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), and **%commonatts;** (C.19) for more on this element's attributes.

Example of providing both SGML and **troff tbl** markup of a table:

```
<Table altreps="BR-SR-3031-tbl11"> ... </Table><ExternalText
id="BR-SR-3031-tbl11" present="altrep" entityref="table11tbl">
```

extra= E.17

External traversal rules. The value of "A A" tells HyTime processors that external traversal of a link (in TIM, a **Link**) may be initiated or continued from either link end.

```
          extra NMTOKENS "A A"
```

The **intra** attribute (I.15), which governs internal traversal, has the same value.

F (FAX — FrontMatter)

"Fault", type= — ListTerm (L.14), T (T.1)

FAX (notation, TIM 1 only) F.1

Notation that should be declared for a `filename.c4` or `filename.cal` bitmap file with CCITT Group 4 compression, sometimes called a CALS Type I bitmap file.

FAX files should not be included in TEDD packages that follow all TCIF Guidelines. Currently the TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG (P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18).

"FAX", type= — T (T.1)

FAXTILE (notation, TIM 1 only) F.2

Notation that should be declared for a `filename.c4` or `filename.cal` tiled bitmap file with CCITT Group 4 compression, sometimes called a CALS Type II bitmap file.

```
<!ENTITY fig001 SYSTEM "../fax/fig001.c4" NDATA FAXTILE>
```

FAXTILE files should not be included in TEDD packages that follow all TCIF Guidelines. Currently the TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG (P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18).

"Feedback", type= — Section (S.4)

"Fieldname", type= — T (T.1)

"Figure", type= — Frame (F.8)

"Figures", type= — Contents (C.22)

FileBasename (notation) F.3

A noncommittal notation name for non-text files.

```
<!NOTATION FileBasename PUBLIC "-//TCIF-IPI-95-004::FileBasename//
NOTATION File
```

It could be used in declarations for external entities when more than one file format is provided and it is left to the recipient to choose a preferred format. For

instance, a document originator might provide both GIF and EPS files for every graphic in a document. Rather than assume that the recipient wants either

```
<!ENTITY fig001 SYSTEM "../gif/fig001.gif" NDATA GIF>
```

which specifies the GIF version, or

```
<!ENTITY fig001 SYSTEM "../eps/fig001.eps" NDATA EPS>
```

the originator could use

```
<!ENTITY fig001 SYSTEM "fig001" NDATA FileBasename>
```

This would let the recipient choose from all provided formats according to a set of preferences. There is a big catch: this notation leaves it to the browser to search the TEDD-specified directories in the preferred order until a file with the supplied basename (filename minus extension) is found. No browser application as yet knows how to do this. However, UNIX- and Windows-based **`perl`** programs have been written to batch-convert FileBasename entity declarations in TIM files into GIF/EPS/etc. declarations by performing a multidirectory search after reading a list of preferences.

"Filename", type= — T (T.1)

FIXED (SGML reserved name) F.4

Reserved name used to indicate that the default value for an attribute is also the only permitted value: it cannot be changed. For example, the **`Ref`** element has a fixed **`HyTime`** value of "clink". This attribute value indicates to a HyTime parser that the **`Ref`** element conforms to the HyTime "clink" architectural form. If the **`HyTime`** value were to be changed for any element, an SGML parser would report an error.

<Float> (TIM 1 only) — present (P.16)

"float", present= — (P.16)

<Flowchart> F.5

An element referencing an external entity that graphically or otherwise represents a flowchart. Use of this element is discouraged: it exists mainly as a placeholder for a future, non-empty element to be defined in an extension to TIM (invoked with the **`%flows;`** entity reference (F.6)). It is assumed that the extension will redefine the **`Flowchart`** element and define new subelements for it, but will not need to redefine other elements of the DTD, some of which are currently allowed to contain **`Flowchart`**. Recipients who encounter a **`Flowchart`** element before the extension has been approved should treat it as if it were a **`Graphic`**. Currently it is defined exactly the same way.

```
<!ELEMENT Flowchart - O EMPTY >
<!ATTLIST Flowchart
```

```
%baseatts;
%identity;
%commonatts;
dimensions NMTOKENS #IMPLIED
placement NMTOKENS #IMPLIED
valign (top|middle|bottom) #IMPLIED
align (left|right|center) #IMPLIED
layer NMTOKEN #IMPLIED
entityref ENTITY #REQUIRED >
```

Use of the **type** attribute is not supported. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), and **%commonatts;** (C.19) for more on this element's attributes.

%flows; F.6

An entity reference defined in TIM to allow new element declarations for flowchart markup to be included. This will be done if the document prologue contains an entity declaration for **%flows;** that points to an actual file. Example:

```
<!ENTITY % flows SYSTEM "../../lib/flowchrt.dtd" NDATA SGML>
```

It is assumed that one of the declarations in the file would be for the element **Flowchart**, and the rest would be for subelements of **Flowchart**. Since TIM reads these declarations (if they exist) before its own declaration for **Flowchart**, the new ones will be in effect. By default, **%flows;** is a null string, so its occurrence in the DTD does nothing:

```
<!ENTITY % flows        "">
%flows;
```

There are seven entity references for extensions to TIM 2, and their placement (before all element declarations in the DTD) and order (**%other;**, **%memos;**, **%rqmts;**, **%procs;**, **%eqns;**, **%flows;**, **%tables;**) are significant, since the order is the priority for redefining elements: only the first element and attribute declarations for a given element are used. The order reflects the likelihood that declarations in one extension would redefine elements in others: the table extension (already provided) is, of course, the least likely to affect others.

"FNote", type= — Annote (A.9)

footnote — Annote (A.9)

"Foreword", type= — Section (S.4)

format= F.7

Indicates the data representation used for an element's contents. Used (with a value such as "HTML", "TBL", "EQN", or "PIC") to specify that the content of an element contains literal text, without TIM markup.

```
format NOTATION (SGML|HTML|RTF|TEX|TBL|EQN|PIC|EPS|PS|PBM|PGM
  |PPM|CGM-CLEAR|TEXT) #IMPLIED
```

This attribute is defined for all elements that have the **%textatts;** (T.12). Default value: undefined, meaning that the notation is TIM SGML. Specifying the notation can be helpful for certain kinds of processing, but it is not enough to force SGML applications to ignore characters that could be SGML markup and to preserve white space and line breaks (usually the preferred behavior for non-SGML data). To get those things to happen, you must

- Use the rendering application's method of identifying "literal" or "preformatted" elements (in TIM, the elements that should be identified that way are **GraphicalText** (G.3) and **Listing** (L.12)), or
- Put the non-SGML text/data in a CDATA marked section (M.2) (precede it with "<!(CDATA)" and follow it with "))>"), or
- Keep it in a separate file and use an entity reference (E.8).

Notations exist for non-text (binary) data, but these can be used only in defining external entities. Binary data cannot be included directly in the SGML file.

"Foreword", type= — Section (S.4)

<Frame> F.8

(**Frame** is also an attribute, F.9.) A holder for figures, tables, equations, listings, and anything else distinct from running text. Use of **Frame** implies that the contents could be presented in a separate box or window (they don't have to be). The content model is that of an **S** structure (S.1), with the added possibilities of **Equation** and **GraphicalText**. The figure/equation/listing number and title are part of the frame, not of the object inside the frame; to identify separate numbering streams (e.g., "Figure 1" vs. "Exhibit 2"), use the **type** attribute (e.g., **<Frame type="Figure">**).

```
<!ELEMENT Frame - - ((Title|TitleGroup)?,(P|S|Annote|Frame|Danger
          |Caution|Warning|Admon|DistOrdLI|DistVarLI|Table|OrderedList
          |UnorderedList|VariableList|SegmentedList|Listing|Pt
          |IndexTerm|Graphic|Object|Flowchart|ExternalText|Equation
          |GraphicalText)*)>
<!ATTLIST Frame
          %baseatts;
          %identity;
          %numbering;
          %commonatts;
          dimensions NMTOKENS #IMPLIED
          placement NMTOKENS #IMPLIED
          valign (top|middle|bottom) #IMPLIED
          align (left|right|center) #IMPLIED
          layer NMTOKEN #IMPLIED >
```

Values of **type** that should be recognized: "Table"; "Graphic"; "Figure"; "Exhibit"; "Icon" (untitled graphic); "Equation"; "Sidebar" (framed text); "Listing" . See the

entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), and **%commonatts;** (C.19) for more on this element's attributes.

Style Note

A **Frame** is an imaginary box around some portion of the content of a document (a figure or table inside a **Frame** may have a real box around it, but that's different), and an effort should be made to keep this box intact on presentation — it may have to float a little from its exact spot in the text stream to fit on a page or screen. But a **Frame** is not inherently a floating object — one that should appear in a separate window on a screen or at the top or bottom of a page on paper. To specify that it should float like that, set its **present** attribute to "float". Authors should not assume that a **Frame** will not have to float — have its position adjusted — at all, since the size of the page or screen is not in their control — so references like "see below" should be avoided.

frame= F.9

(**Frame** is also an element, F.8.) Applies to **Table** only. Specifies where rule lines should appear on the outer margins of the table.

```
frame (top|bottom|topbot|all|sides|none) #IMPLIED
```

There is no default, so the rendering application must supply one.

<FrontMatter> F.10

An optional major subdivision of **TDoc** (T.9), containing any **Sections** or contents lists before the **Body** sections. It exists, along with the other major subdivisions (required **Body**, optional **AppMatter**, and optional **BackMatter**) mainly to aid applications in identifying and numbering the sections within it.

```
<!ELEMENT FrontMatter - - ((Title|TitleGroup)?,(Contents|Index|P|S
          |Annote|Frame|Danger|Caution|Warning|Admon|DistOrdLI|DistVarLI
          |Table|OrderedList|UnorderedList|VariableList|SegmentedList
          |Listing|Pt|IndexTerm|Graphic|Object|Flowchart|ExternalText
          |Section)*) >
<!ATTLIST FrontMatter
          %baseatts;
          %identity;
          alerts IDREFS #IMPLIED
          altreps IDREFS #IMPLIED >
```

See the entries for the entities **%baseatts;** (B.2) and **%identity;** (I.3) for details on this element's attributes. Use of the **type** attribute is not currently supported (no standard values have been established).

G (GIF — groups)

GIF (notation) G.1

Notation that should be declared for a `filename.gif` CompuServe GIF (Graphic Interchange Format) bitmap file.

```
<!NOTATION GIF       PUBLIC "+//ISBN 0-7923-9432-1::Graphic
           Notation//NOTATION CompuServe Graphic Interchange
           Format//EN">
```

Example:

```
<!ENTITY fig001 SYSTEM "../gif/fig001.gif" NDATA GIF>
```

The file should be stored in a `gif` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s). Currently the TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG (P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18).

GIF files should be in GIF87a format and should be 4-bit (16-color) or 8-bit (256-color) only. The later GIF89a format and other bit depths are not read by as many applications. (GIF animation, a feature of GIF89a format, is considered a multimedia format; acceptable multimedia formats will be the subject of a future TCIF Guideline.) The version can be read in the first six bytes of the file: they are either "GIF87a" or "GIF89a". GIF uses a patented compression method that software vendors (not users) must pay royalties for, so there is some motivation to replace GIF with the PNG format, but so far PNG is supported by far fewer applications than GIF. JPEG (JFIF) format has already largely replaced GIF for photographic images, because there is no 24-bit ("true-color") form of GIF. Compression in GIF format is typically about 65% with photographic images, so files can still be very large, but diagrams that are mostly white background can be compressed more than 95%.

"Glossary", type= — Section (S.4), Index **(I.9)**

<GlossEntry> (TIM 1 only)

Dropped from TIM 2 as unnecessary. Use a **`VariableList`** (V.2).

"GlossTerm", type= — ListTerm (L.14) and T **(T.1)**

<Graphic> G.2

An element referencing an external graphic entity. A **`Graphic`** element always references an external graphic image (GIF, CGM, EPS, etc.) with a simple rectangular, 2-dimensional shape; other non-ASCII files (3-D images, audio,

multimedia, encrypted text, etc.) should be referenced by an **Object** element (O.1).

```
<!ELEMENT Graphic - O EMPTY>
<!ATTLIST Graphic
          %baseatts;
          %identity;
          %commonatts;
          dimensions NMTOKENS #IMPLIED
          placement NMTOKENS #IMPLIED
          valign (top|middle|bottom) #IMPLIED
          align (left|right|center) #IMPLIED
          layer NMTOKEN #IMPLIED
          entityref ENTITY #REQUIRED>
```

Use of the **type** attribute is not supported. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), and **%commonatts;** (C.19) for more on this element's attributes.

"Graphic", type= — Frame (F.8)

<GraphicalText> G.3

An element containing text characters that draw a picture (the characters may include entity references to special graphical characters). It may occur in-line in running text or in a **Frame** (F.8), or conceivably elsewhere. It has the data-content model of a **T** text phrase (T.1). It is not numbered itself, but an enclosing **Frame** can be.

A **GraphicalText** element should be assumed to be "preformatted" — white space and line breaks should be preserved, but entity references and subelements recognized.

```
<!ELEMENT GraphicalText - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef
          |IndexTerm|Pt|Graphic|Object|ExternalText|SimpleList|Equation
          |GraphicalText)* >
<!ATTLIST GraphicalText
          %baseatts;
          %identity;
          %commonatts;
          %textatts; >
```

Use of the **type** attribute is not supported. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%commonatts;** (C.19), and **&textatts;** (T.12) for more on this element's attributes.

A typical graphic made with **GraphicalText** might be marked up this way:

```
<P>This is a Smiley: <GraphicalText>:-)</GraphicalText></P>
```

Comment

GraphicalText is not intended for all portions of text that are commonly rendered in constant-width type (filenames, **input** and output strings, etc.) — those should be marked as T phrases with the appropriate type value. Use **GraphicalText** only for characters meant to draw a picture.

group= G.4

Indicates that an element belongs to the element group(s) identified by the attribute. **Group** is an arbitrary grouping, and implies nothing about formatting, structure, or meaning, as **type** usually would.

```
group NMTOKENS #IMPLIED
```

The default value is undefined. Definitions of values are left to the document originator. Although the attribute value can be more than one NMTOKEN, multiple values would be difficult for many applications to process.

This attribute is defined for all elements that have the **%baseatts;** (B.2).

Comment

Since there are neither rules nor conventions on how to use this attribute, it should contain only information the originating and receiving organizations have agreed upon, or information for the originator's internal use only. Originators should expect that most recipients will ignore the attribute.

groups= G.5

Specifies values of the **group** attribute (G.4) that elements should have in order to be listed in a **Contents** list or an **Index**. See **Contents** (C.22) for more explanation.

```
groups NMTOKENS #IMPLIED
```

The default value of this attribute is undefined: all values of **group** should be included.

H (HTML — HyTime)

"History", type= — Section (S.4)

"holds', continuation= (C.23)

"horiz', arrange= (A.12)

"How-To-Use", type= — Section (S.4)

HTML (notation) H.1

A text notation containing HTML (HyperText Markup Language) markup.

```
<!NOTATION HTML      SYSTEM "HTML">
```

It may be within a TIM file:

```
<Listing format="HTML"><HTML> ....
```

Or it may be in a separate file with a `.htm` extension, in an `html` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s):

```
<!ENTITY html001 SYSTEM "../html/html001.htm" NDATA HTML>
...
<ExternalText entityref="html001">
```

HyNames= H.2

This attribute allows a HyTime parser to recognize another attribute as a particular HyTime attribute, when its name is different from the standard HyTime name. When used in TIM, **`HyName`** relates the name of the IDREF attribute to the standard HyTIme name "linkend." The value of this attribute is always fixed and should not be modified.

For **`Ref`** (R.2), the definition is:

```
HyNames CDATA #FIXED "linkend rid"
```

For **`Pt`**, the definition is:

```
HyNames CDATA #FIXED "linkend spanend"
```

HyTime= H.3

This attribute allows a HyTime parser to recognize an element as having the specified architectural form. The NAME value of this attribute is always fixed and should not be modified. **`NameLoc`** (N.3), **`NmList`** (N.3), **`DataLoc`** (D.2), and **`TreeLoc`** (T.22), which exist only for HyTime use, were named for their HyTime forms, so the value of **`HyTime`** for each is the same as its name. For instance:

```
HyTime NMTOKEN #FIXED "NameLoc"
```

Link (L.9), when used for HyTime linking, fits the architectural form of an ILink:

```
HyTime NMTOKEN #FIXED "ilink"
```

Ref (R.2) and **Pt** (P.23) fit the architectural form of a CLink:

```
HyTime NMTOKEN #FIXED "clink"
```

?HyTime (processing instructions) H.4

Four processing instructions that identify the version of HyTime and the particular modules of the HyTime standard that are implemented in TIM:

```
<?HyTime VERSION "ISO/IEC 10744:1992" HYQCNT=32>
<?HyTime MODULE base>
<?HyTime MODULE locs>
<?HyTime MODULE links>
```

I (id — ISOtech)

"Icon", type= — Frame (F.8)

id= I.1

Identifier. Provides an element with an identifier (or "name") that is unique at least within the scope of the current document. When required, it may be system-generated. The definition of **id** for the HyTime elements **NameLoc** (N.3), **TreeLoc** (T.22), and **DataLoc** (D.2) is:

```
id ID #REQUIRED
```

The definition for other elements is:

```
id ID #IMPLIED
```

Although the definition does not require an **id** for most elements, any element that is pointed to by a **Link** or **Ref** must have an **id**.

This attribute is usually defined through the entity reference **%identity;** (I.3).

IDs are SGML NAMEs, for which TIM specifies a length limit of 64 characters. Allowed characters are A-Z, 0-9, hyphen ("-"), and period ("."), and only A-Z may be the first character. (Lowercase is allowed, but is treated as uppercase.)

Comment

The value of ID will often be system-generated, but it could be systematically assigned to assure a unique identifier across all telecom documents, by concatenating the unique code for the originating company, the document identifier (from **DN.base**), and a sequentially assigned number within the document. The use of such unique IDs is recommended. The ID might also incorporate an ID type indicator (ALERT, TABLE, FIGURE, INDEX, etc.) to guide users when there are multiple cross-references.

Example:

```
<Frame type="Table" id="TCIF-IPI-95-004-TABLE-1" ...>
```

One highly recommended use of type indicators is for HyTime-conforming external references using **Link** or **Ref**. Since processing is more complicated for external references, it is wise to signal whether the cross-reference uses **NameLoc** to access an external file:

```
<NameLoc id="TCIF-IPI-95-004-NAMELOC-1" ...>
```

ID (SGML reserved name) I.2

Reserved name for a unique identifier for an element (**id** is also an attribute, see I.1). IDs follow the rules for NAMEs (N.1), plus one more: each must be unique

within the document. Attributes that have a value type of ID are used to uniquely identify elements so that they may be pointed to by other elements:

```
<Section label="1" id="BR-123-456-789-X12345">...</Section>
```

In this example, Section 1 has a unique identifier of "BR-123-456-789-X12345" that can be referenced by other elements.

"ID", type= — T (T.1)

%identity; I.3

An entity defining identifying attributes available for many elements, but deliberately unavailable for title elements to ensure that they will instead be used on the sections, tables, etc., that have the titles:

```
<!ENTITY % identity
         "id ID #IMPLIED
          label CDATA #IMPLIED
          keywords CDATA #IMPLIED">
```

Whenever "%identity;" appears in an element's ATTLIST, the three attribute declarations above are substituted for it. Refer to the individual entries for the meanings and uses of each: **id** (I.1), **label** (L.1), and **keywords** (K.2).

IDREF (SGML reserved name) I.4

Reserved name for a reference to an ID. An IDREF has the same data content as a NAME; however, it must exactly match the value of a declared ID somewhere in the same document. Attributes (such as **rid**, R.18) that have a value type of IDREF are used to point to other elements with matching ID values. This **Section** has an ID of "BR-123-456-789-X12345" that uniquely identifies it:

```
<Section label="1" id="BR-123-456-789-X12345">...</Section>
```

If a **Ref** cross-reference were to point to it, it would have an **rid** attribute whose value was an IDREF that matched the **Section**'s ID:

```
... as shown in <Ref rid="BR-123-456-789-X12345">Section 1</
Ref> of this document.
```

IDREFS (SGML reserved name) I.5

Reserved name for a list of IDREFs; with each IDREF separated by white space. An IDREFS attribute has the same data content as a NAMES attribute; however, each name must exactly match the value of a declared ID somewhere in the same document. Attributes (such as **alerts** and **altreps**) that have a value type of IDREFS can be used to point to more than one target element.

"ignore", use= (U.5)

IMPLIED (SGML reserved name) I.6

Reserved name that can be used in place of a default value for an attribute. An implied attribute is one whose value does not need to be explicitly specified in the document. An attribute value may be implied for one of three reasons:

1. The attribute is optional and is not required for the element to be processed.
2. The attribute has a default value that should be implicitly understood by the rendering application and, therefore, does not need to be specified.
3. The attribute value is to be inserted automatically by a processing system (as with autonumbering of list items).

Although it is not mandatory to specify a value for an attribute with an IMPLIED default value, it is necessary if the desired attribute value is different from the implied default (for example, if the implied default value of **emph** is "ital" but the emphasis is intended to be "bold").

"Important", type= — Admon (A.2) and Annote **(A.9)**

include= I.7

Specifies what content should be generated for a **Contents** list, **Index**, or **Ref**. See **Contents** (C.22) for more explanation.

```
include NMTOKENS #IMPLIED
```

The default value of this attribute is undefined — that is, no content should be generated. Values that rendering applications should understand are: **num** (element number or mark), **title** (element title), **text** (element text), **allcontent** (element content of all forms), **page** (page number), and **link** (hypertext link from contents/index list item to element).

"include", use= — (U.5)

inclusions, in element declarations — ELEMENT (E.3)

increment= I.8

For numbered elements, **increment** specifies how much greater or smaller the next number in a sequence should be. Of course, the default is "1", for numbering 1, 2, 3, ... (or A, B, C ... or i, ii, iii ...). A value of "-1" should yield the sequence like 5, 4, 3, ... (depending on the first number, set with **restartsat**, R.15). This attribute is not for specifying which items in a sequence (say, every 3rd or 5th) have a visible number; use **shownum** for that.

```
increment NMTOKEN #IMPLIED
```

The value must be an unsigned or negative integer ("-1", "2", etc.). Only the value "-1" is likely to be used, if the attribute is ever used at all, and document providers should anticipate that many recipients will not be able to implement even that with current applications.

The **increment** attribute is defined, for the elements that have it, through the entity reference **%numbering;** (N.13).

<Index> I.9

An alphanumerically sorted list of contents, or a specification for one. May appear within the major subdivisions of **TDoc** or any level of **Section**, and may be a typical index (listing key words and phrases, which could be identified in the text with **IndexTerm** elements) or an alphabetized list of some other specified element. It is not required to be at the end of a Section. It may (should?) contain hypertext links rather than page numbers. Its content model and attributes are exactly the same as those of **Contents** (C.22) — the difference is entirely in the ordering of the list: A **Contents** list has items in the order they appear in the document, while an **Index** lists items in alphanumeric order.

To specify how the index list is to be constructed, special attributes are available. They are the same as for **Contents**. Please refer to the description there (C.22).

Index has a **sortorder** attribute (S.17) to specify how nonalphabetic characters should be sorted.

```
<!ELEMENT Index - - ((Title|TitleGroup)?,(P|S|Annote|Frame|Danger
          |Caution|Warning|Admon|Table|OrderedList|UnorderedList
          |VariableList|SegmentedList|Listing|Pt|IndexEntry|Graphic
          |Object|ExternalText|Flowchart|Index)*) >
<!ATTLIST Index
          %baseatts;
          %identity;
          sortorder CDATA #IMPLIED
          elements NMTOKENS #IMPLIED
          levels NMTOKENS #IMPLIED
          groups NMTOKENS #IMPLIED
          types NMTOKENS #IMPLIED
          remaps CDATA #IMPLIED
          langs CDATA #IMPLIED
          revstats NMTOKENS #IMPLIED
          statuses CDATA #IMPLIED
          scope NMTOKENS #IMPLIED
          include NMTOKENS #IMPLIED >
```

Values of **type** that should be recognized: "Name", "Subject", "Glossary" (for a glossary generated from **IndexTerms**), "Procedures", "RqmtObjs" (requirement objects), and "Index" (an all-in-one index, the default). See the entries for the entities **%baseatts;** (B.2) and **%identity;** (I.3) for more on this element's attributes.

Comment

An author who does not wish to make assumptions about the capabilities of the rendering application could include both specifications (in the form of attribute values) and content for a **Index** element. Typically, an unsophisticated rendering application would present the supplied content, while a more capable one would generate content matching the specifications. However, rendering applications are free to ignore both the supplied content and the specifications, and even the presence or absence of a **Index** element, for the sake of consistency of presentation across a document set. Authors should particularly consider the futility of supplying page numbers when the format and medium of presentation are indeterminate. (For almost all other elements, only formatting instructions, such as numbering, may be ignored; content may not be.)

"Index", type= — Index (I.9) and IndexTerm (I.11)

<IndexEntry> I.10

Either:

- An entry or part of an entry in an index or contents list, normally with links to the places where the topic occurs in the body text.
- Part of the raw material for such an entry, located within the target element that the generated entry will point to. In this case the complete entry is enclosed in **IndexTerm** rather than **IndexEntry** to signal that the content should not be displayed (it should be processed into an entry somewhere else).

IndexEntrys in a **Contents** or **Index** should contain cross-references to other elements; the target can be any element with an ID. There can be multiple targets for an entry, requiring multiple references. These references would appear one at a time in **Ref** elements.

An **IndexEntry** can include any number of nested **IndexEntry**s, and the nesting can be to any level, to produce hierarchical entries. This tertiary entry —

Indexes
creating
in TIM markup, I.10

— would be created from this markup:

```
<IndexEntry>Indexes<IndexEntry>creating<IndexEntry>in TIM
markup, <Ref rid="TCIF-IPI-95-004-INDEX-5679">I.10</Ref></
IndexEntry></IndexEntry></IndexEntry>
```

The content model of **IndexEntry** allows nested **IndexEntry**s to appear anywhere within them, to be compatible with XML's restrictions on element content models, but in practice they should appear only after all content at the current level of nesting. That is, when **IndexEntry**s are nested, there should

never be anything after their end-tags except a sibling's start-tag or the parent's end-tag: either **</IndexEntry><IndexEntry>** or **</IndexEntry></IndexEntry>**.

```
<!ELEMENT IndexEntry - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef
          |Equation|GraphicalText|IndexEntry|Pt)* >
<!ATTLIST IndexEntry
          %baseatts;
          %commonatts;
          %textatts;
          id ID #IMPLIED
          prefer (preferred|ordinary) #IMPLIED >
```

Use of the **type** attribute is not supported (carrying over the type value of an **IndexTerm** would be harmless).The **prefer** attribute is meant for **IndexEntry**s nested in **IndexTerm**s (I.11); it indicates the best reference in the text from a given entry in the index, the one the reader's attention should be directed to. It might be indicated by making the **Ref** created for the actual index entry bold.

See the entries for the entities **%baseatts;** (B.2), **%commonatts;** (C.19), and **&textatts;** (T.12) for more on attributes of **IndexEntry**.

Style Note

For electronic deliverables, page-number references should be avoided as much as possible. Section numbers are preferable, but hyperlinked index need not contain any kind of number, as long as there is an obvious hot zone to click.

<IndexTerm> I.11

An element meant to mark the cross-reference target location of, and provide the content for, an index entry that will be generated in some later processing step. This step consists of at least:

1. Moving the element to its appropriate place (alphanumerically) in an appropriate **Index** (if there is more than one index, the **IndexTerm**'s **type** attribute should say which one it belongs in).
2. Changing the **IndexTerm** to an **IndexEntry** (the **IndexTerm** may contain nested **IndexEntry**s that don't change).
3. Creating a cross-reference (using a **Ref** element), if not already present, to the original parent element of the **IndexTerm** and placing it at the end of the most deeply nested **IndexEntry**.

A more sophisticated application would finish the process by combining the individual index entries that have the same content for each hierarchical level into a single compound entry. That is, these:

```
<IndexEntry>Indexes<IndexEntry>creating<IndexEntry>using
IndexEntry, <Ref rid="TCIF-IPI-95-004-INDEX-5679">I.10</Ref></
IndexEntry></IndexEntry></IndexEntry>

...

<IndexEntry>Indexes<IndexEntry>creating<IndexEntry>using
IndexTerm, <Ref rid="TCIF-IPI-95-004-INDEX-5680">I.11</Ref></
IndexEntry></IndexEntry></IndexEntry>
```

would become:

Indexes
  creating
    with IndexEntry, I.10
    with IndexTerm, I.11

The text within the original **IndexTerm** should not be rendered where it appears; that is, it does not do double duty as both index content and body content. This means that typically the term appearing in the **IndexTerm** will also appear immediately outside it. That word or phrase can be wrapped, along with the **IndexTerm**, in a **T** element that can serve as the target for cross-reference from the index:

```
... <T id="XXX">Indexes<IndexTerm>Indexes</IndexTerm></T> can
be created ...
```

Otherwise, the first ancestor element of the **IndexTerm** to have an ID should serve as the target. See **IndexEntry** for more explanation.

```
<!ELEMENT IndexTerm - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef
          |Equation|GraphicalText|IndexEntry|Pt)* >
<!ATTLIST IndexTerm
          %baseatts;
          %commonatts;
          %textatts;
          id ID #REQUIRED
          prefer (preferred|ordinary) #IMPLIED >
```

If there is more than one index, values of the **type** attribute should be used to indicate which one the **IndexTerm** belongs in. Values that should be recognized: "Name", "Subject", "Glossary" (for a glossary generated from **IndexTerm**s, which is theoretically possible), and "Index" (an all-in-one index, the default). The **prefer** attribute indicates the best reference in the text from a given entry in the index, the one the reader's attention should be directed to. It might be indicated by making the **Ref** created for the actual index entry bold.

See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%commonatts;** (C.19), and **&textatts;** (T.12) for more on this element's attributes.

Comment

There are at least two reasons to include floating index entries rather than a formatted index. One is that it is easier to maintain the entries if they are stored next to the points in the text that they reference. The other is that it leaves generation of the content of the cross-references to the rendering application, which can insert page numbers for paper and something else for electronic use. The reason to include a formatted index instead is to save the recipient the trouble of generating it. No tools exist at the time of this writing to automate the generation of a TIM index.

IndexText (notation, TIM 1 only) —

Removed from TIM 2 as unnecessary. Use TEXT (T.11).

infix= I.12

An optional separator for concatenated numbers of a section, list item, frame, paragraph, etc. — for instance, the period in "Section 1.2" or the hyphen in "Figure 2-1."

```
infix CDATA #IMPLIED
```

The **infix** attribute is defined, for the elements that have it, through the entity reference **%numbering;** (N.13).

The default value is undefined — a rendering application must be prepared to choose the character(s) used when **inheritnum** is used (I.13). Any supplied value should be ignored if **inheritnum** is not set to "inherit ...". When multiple levels of ancestor numbers are inherited, as in "Section 1.4.3.5," the value of **infix** applies only to the last of the separators. The earlier separators should have been determined at the higher-level elements.

Since some rendering applications can't distinguish a null attribute value, "", from an undefined one, the special value "noinfix" should be interpreted as an explicit "", meaning do use no infix rather than choosing a different default.

"Informal", type= — SubTitle (S.27) and SuperTitle **(S.32)**

"inherit", "inherit1", "inherit2", "inherit3", inheritnum= (I.14)

inheritfrom= I.13

When **inheritnum** (I.13) is specified, **inheritfrom** can be used to specify an element type or types other than the current element from which to inherit numbers.

```
inheritfrom NMTOKENS #IMPLIED
```

The **inheritfrom** attribute is defined, for the elements that have it, through the entity reference **%numbering;** (N.13).

The default value is the current element type. If the number of names listed is less than the number of levels specified by **inheritnum**, the last name specified applies to all lower levels. See examples under **inheritnum**.

inheritnum= I.14

Specifies whether the number of a section, list item, frame, paragraph, etc., should have the numbers of higher-level elements concatenated with it — for instance, "2.4.1" for a third-level section. The value "inherit1" specifies that only the highest-level element's number should be inherited — for instance, the number of the current first-level **Section**, but not the numbers of lower-level **Section**s, as in "Figure 3-1." Similarly, "inherit2" says inherit two numbers and "inherit3" three numbers (we've never seen that done, but the value is defined just in case). "Inherit" means inherit as many levels as there are. "Inherit0" allows the **infix** attribute's value to be included in the label without an inherited number.

```
inheritnum (inherit|inherit0|inherit1|inherit2
  |inherit3|noinherit) #IMPLIED
```

Default value: "noinherit". The **inheritnum** attribute is defined, for the elements that have it, through the entity reference **%numbering;** (N.13).

If **inheritfrom** is also specified, numbers should be inherited from the element(s) specified rather than the same type of element in which **inheritnum** appears. Examples:

- To specify that a **Section** inherits the numbers of all higher-level **Section**s ("1.2.4.8.1"), use **<Section inheritnum="inherit" infix=".">**
- To specify that tables inherit the number of the highest-level **Section** ("Table 3-1"), use **<Frame type="Table" inheritnum="inherit1" inheritfrom="Section" infix="-">**
- To specify that figures within a requirement object (marked as a **DistVarLI**) inherit the requirement number, which already inherits the number of the highest-level **Section** ("Figure 3-4-1"), use **<Frame type="Figure" inheritnum="inherit2" inheritfrom="Section DistVarLI" infix="-">**.

"inline", arrange= (A.12)

"Input", type= — Listing (L.12), T (T.1)

"insert", revstat= (R.16)

"insflag", revstat= (R.16)

intra= I.15

Internal traversal rules. The value of "A A" tells HyTime processors that internal traversal of a link (in TIM, a **Link**) may be initiated or continued from either link end.

```
intra NMTOKENS "A A"
```

The **extra** attribute (E.17), which governs external traversal, has the same value.

<ISN> I.16

An optional subelement of **DocID** (D.27) supplying the document's ISBN or ISSN (International Standard Book or Serial Number), if it has one.

```
<!ELEMENT ISN    - -  (#PCDATA)* >
<!ATTLIST ISN
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Examples:

```
<ISN>ISBN 0-19-853737-9</ISN>

<ISN>ISSN 0010-7174</ISN>
```

%ISObox; I.17

An entity reference to a set of ISO standard character entity references for box-drawing characters, which can be mapped, for one instance, to characters 179 through 218 in the IBM PC DOS font:
³´µ¶·¸¹º»¼½¾¿ÀÁÂÃÄÅÆÇÈÉÊËÌÍÎÏÐÑÒÓÔÕÖ×ØÙÚ.

```
<!ENTITY % ISObox PUBLIC "ISO 8879-1986//ENTITIES Box and Line Drawing//
EN">
%ISObox;
```

This is the entity set:

```
<!-- (C) International Organization for Standardization 1986
     Permission to copy in any form is granted for use with
```

```
     conforming SGML systems and applications as defined in
     ISO 8879, provided this notice is included in all copies.-->
<!-- All names are in the form: box1234, where:
    box = constants that identify a box drawing entity.
    1&2 = v, V, u, U, d, D, Ud, or uD, as follows:
      v = vertical line for full height.
      u = upper half of vertical line.
      d = downward (lower) half of vertical line.
    3&4 = h, H, l, L, r, R, Lr, or lR, as follows:
      h = horizontal line for full width.
      l = left half of horizontal line.
      r = right half of horizontal line.
    In all cases, an upper-case letter means a double or heavy line.-->
<!ENTITY boxh   SDATA "[boxh  ]"--horizontal line -->
<!ENTITY boxv   SDATA "[boxv  ]"--vertical line-->
<!ENTITY boxur  SDATA "[boxur ]"--upper right quadrant-->
<!ENTITY boxul  SDATA "[boxul ]"--upper left quadrant-->
<!ENTITY boxdl  SDATA "[boxdl ]"--lower left quadrant-->
<!ENTITY boxdr  SDATA "[boxdr ]"--lower right quadrant-->
<!ENTITY boxvr  SDATA "[boxvr ]"--upper and lower right quadrants-->
<!ENTITY boxhu  SDATA "[boxhu ]"--upper left and right quadrants-->
<!ENTITY boxvl  SDATA "[boxvl ]"--upper and lower left quadrants-->
<!ENTITY boxhd  SDATA "[boxhd ]"--lower left and right quadrants-->
<!ENTITY boxvh  SDATA "[boxvh ]"--all four quadrants-->
<!ENTITY boxvR  SDATA "[boxvR ]"--upper and lower right quadrants-->
<!ENTITY boxhU  SDATA "[boxhU ]"--upper left and right quadrants-->
<!ENTITY boxvL  SDATA "[boxvL ]"--upper and lower left quadrants-->
<!ENTITY boxhD  SDATA "[boxhD ]"--lower left and right quadrants-->
<!ENTITY boxvH  SDATA "[boxvH ]"--all four quadrants-->
<!ENTITY boxH   SDATA "[boxH  ]"--horizontal line-->
<!ENTITY boxV   SDATA "[boxV  ]"--vertical line-->
<!ENTITY boxUR  SDATA "[boxUR ]"--upper right quadrant-->
<!ENTITY boxUL  SDATA "[boxUL ]"--upper left quadrant-->
<!ENTITY boxDL  SDATA "[boxDL ]"--lower left quadrant-->
<!ENTITY boxDR  SDATA "[boxDR ]"--lower right quadrant-->
<!ENTITY boxVR  SDATA "[boxVR ]"--upper and lower right quadrants-->
<!ENTITY boxHU  SDATA "[boxHU ]"--upper left and right quadrants-->
<!ENTITY boxVL  SDATA "[boxVL ]"--upper and lower left quadrants-->
<!ENTITY boxHD  SDATA "[boxHD ]"--lower left and right quadrants-->
<!ENTITY boxVH  SDATA "[boxVH ]"--all four quadrants-->
<!ENTITY boxVr  SDATA "[boxVr ]"--upper and lower right quadrants-->
<!ENTITY boxHu  SDATA "[boxHu ]"--upper left and right quadrants-->
<!ENTITY boxVl  SDATA "[boxVl ]"--upper and lower left quadrants-->
<!ENTITY boxHd  SDATA "[boxHd ]"--lower left and right quadrants-->
<!ENTITY boxVh  SDATA "[boxVh ]"--all four quadrants-->
<!ENTITY boxuR  SDATA "[boxuR ]"--upper right quadrant-->
<!ENTITY boxUl  SDATA "[boxUl ]"--upper left quadrant-->
<!ENTITY boxdL  SDATA "[boxdL ]"--lower left quadrant-->
<!ENTITY boxDr  SDATA "[boxDr ]"--lower right quadrant-->
<!ENTITY boxUr  SDATA "[boxUr ]"--upper right quadrant-->
<!ENTITY boxuL  SDATA "[boxuL ]"--upper left quadrant-->
<!ENTITY boxDl  SDATA "[boxDl ]"--lower left quadrant-->
<!ENTITY boxdR  SDATA "[boxdR ]"--lower right quadrant-->
```

%ISOgrk3; I.18

An entity reference to a set of ISO standard character entity references for Greek characters (when used as symbols rather than in Greek text).

```
<!ENTITY % ISOgrk3 PUBLIC "ISO 8879-1986//ENTITIES Greek Symbols//EN">
%ISOgrk3;
```

This is the entity set:

```
<!-- (C) International Organization for Standardization 1986
     Permission to copy in any form is granted for use with
     conforming SGML systems and applications as defined in
     ISO 8879, provided this notice is included in all copies.-->
<!ENTITY alpha    SDATA "[alpha ]"--=small alpha, Greek-->
<!ENTITY beta     SDATA "[beta  ]"--=small beta, Greek-->
<!ENTITY gamma    SDATA "[gamma ]"--=small gamma, Greek-->
<!ENTITY Gamma    SDATA "[Gamma ]"--=capital Gamma, Greek-->
<!ENTITY gammad   SDATA "[gammad]"--/digamma-->
<!ENTITY delta    SDATA "[delta ]"--=small delta, Greek-->
<!ENTITY Delta    SDATA "[Delta ]"--=capital Delta, Greek-->
<!ENTITY epsi     SDATA "[epsi  ]"--=small epsilon, Greek-->
<!ENTITY epsiv    SDATA "[epsiv ]"--/varepsilon-->
<!ENTITY epsis    SDATA "[epsis ]"--/straightepsilon-->
<!ENTITY zeta     SDATA "[zeta  ]"--=small zeta, Greek-->
<!ENTITY eta      SDATA "[eta   ]"--=small eta, Greek-->
<!ENTITY thetas   SDATA "[thetas]"--straight theta-->
<!ENTITY Theta    SDATA "[Theta ]"--=capital Theta, Greek-->
<!ENTITY thetav   SDATA "[thetav]"--/vartheta - curly or open theta-->
<!ENTITY iota     SDATA "[iota  ]"--=small iota, Greek-->
<!ENTITY kappa    SDATA "[kappa ]"--=small kappa, Greek-->
<!ENTITY kappav   SDATA "[kappav]"--/varkappa-->
<!ENTITY lambda   SDATA "[lambda]"--=small lambda, Greek-->
<!ENTITY Lambda   SDATA "[Lambda]"--=capital Lambda, Greek-->
<!ENTITY mu       SDATA "[mu    ]"--=small mu, Greek-->
<!ENTITY nu       SDATA "[nu    ]"--=small nu, Greek-->
<!ENTITY xi       SDATA "[xi    ]"--=small xi, Greek-->
<!ENTITY Xi       SDATA "[Xi    ]"--=capital Xi, Greek-->
<!ENTITY pi       SDATA "[pi    ]"--=small pi, Greek-->
<!ENTITY piv      SDATA "[piv   ]"--/varpi-->
<!ENTITY Pi       SDATA "[Pi    ]"--=capital Pi, Greek-->
<!ENTITY rho      SDATA "[rho   ]"--=small rho, Greek-->
<!ENTITY rhov     SDATA "[rhov  ]"--/varrho-->
<!ENTITY sigma    SDATA "[sigma ]"--=small sigma, Greek-->
<!ENTITY Sigma    SDATA "[Sigma ]"--=capital Sigma, Greek-->
<!ENTITY sigmav   SDATA "[sigmav]"--/varsigma-->
<!ENTITY tau      SDATA "[tau   ]"--=small tau, Greek-->
<!ENTITY upsi     SDATA "[upsi  ]"--=small upsilon, Greek-->
<!ENTITY Upsi     SDATA "[Upsi  ]"--=capital Upsilon, Greek-->
<!ENTITY phis     SDATA "[phis  ]"--/straightphi - straight phi-->
<!ENTITY Phi      SDATA "[Phi   ]"--=capital Phi, Greek-->
<!ENTITY phiv     SDATA "[phiv  ]"--/varphi - curly or open phi-->
<!ENTITY chi      SDATA "[chi   ]"--=small chi, Greek-->
<!ENTITY psi      SDATA "[psi   ]"--=small psi, Greek-->
<!ENTITY Psi      SDATA "[Psi   ]"--=capital Psi, Greek-->
<!ENTITY omega    SDATA "[omega ]"--=small omega, Greek-->
```

```
<!ENTITY Omega    SDATA "[Omega ]"--=capital Omega, Greek-->
```

%ISOlat1; I.19

An entity reference to a set of ISO standard character entity references for accented Latin characters used in western European languages.

```
<!ENTITY % ISOlat1 PUBLIC "ISO 8879-1986//ENTITIES Added Latin 1//EN">
%ISOlat1;
```

This is the entity set:

```
<!-- (C) International Organization for Standardization 1986
     Permission to copy in any form is granted for use with
     conforming SGML systems and applications as defined in
     ISO 8879, provided this notice is included in all copies.-->
<!ENTITY aacute SDATA "[aacute]"--=small a, acute accent-->
<!ENTITY Aacute SDATA "[Aacute]"--=capital A, acute accent-->
<!ENTITY acirc  SDATA "[acirc ]"--=small a, circumflex accent-->
<!ENTITY Acirc  SDATA "[Acirc ]"--=capital A, circumflex accent-->
<!ENTITY agrave SDATA "[agrave]"--=small a, grave accent-->
<!ENTITY Agrave SDATA "[Agrave]"--=capital A, grave accent-->
<!ENTITY aring  SDATA "[aring ]"--=small a, ring-->
<!ENTITY Aring  SDATA "[Aring ]"--=capital A, ring-->
<!ENTITY atilde SDATA "[atilde]"--=small a, tilde-->
<!ENTITY Atilde SDATA "[Atilde]"--=capital A, tilde-->
<!ENTITY auml   SDATA "[auml  ]"--=small a, dieresis or umlaut mark-->
<!ENTITY Auml   SDATA "[Auml  ]"--=capital A, dieresis or umlaut mark-->
<!ENTITY aelig  SDATA "[aelig ]"--=small ae diphthong (ligature)-->
<!ENTITY AElig  SDATA "[AElig ]"--=capital AE diphthong (ligature)-->
<!ENTITY ccedil SDATA "[ccedil]"--=small c, cedilla-->
<!ENTITY Ccedil SDATA "[Ccedil]"--=capital C, cedilla-->
<!ENTITY eth    SDATA "[eth   ]"--=small eth, Icelandic-->
<!ENTITY ETH    SDATA "[ETH   ]"--=capital Eth, Icelandic-->
<!ENTITY eacute SDATA "[eacute]"--=small e, acute accent-->
<!ENTITY Eacute SDATA "[Eacute]"--=capital E, acute accent-->
<!ENTITY ecirc  SDATA "[ecirc ]"--=small e, circumflex accent-->
<!ENTITY Ecirc  SDATA "[Ecirc ]"--=capital E, circumflex accent-->
<!ENTITY egrave SDATA "[egrave]"--=small e, grave accent-->
<!ENTITY Egrave SDATA "[Egrave]"--=capital E, grave accent-->
<!ENTITY euml   SDATA "[euml  ]"--=small e, dieresis or umlaut mark-->
<!ENTITY Euml   SDATA "[Euml  ]"--=capital E, dieresis or umlaut mark-->
<!ENTITY iacute SDATA "[iacute]"--=small i, acute accent-->
<!ENTITY Iacute SDATA "[Iacute]"--=capital I, acute accent-->
<!ENTITY icirc  SDATA "[icirc ]"--=small i, circumflex accent-->
<!ENTITY Icirc  SDATA "[Icirc ]"--=capital I, circumflex accent-->
<!ENTITY igrave SDATA "[igrave]"--=small i, grave accent-->
<!ENTITY Igrave SDATA "[Igrave]"--=capital I, grave accent-->
<!ENTITY iuml   SDATA "[iuml  ]"--=small i, dieresis or umlaut mark-->
<!ENTITY Iuml   SDATA "[Iuml  ]"--=capital I, dieresis or umlaut mark-->
<!ENTITY ntilde SDATA "[ntilde]"--=small n, tilde-->
<!ENTITY Ntilde SDATA "[Ntilde]"--=capital N, tilde-->
<!ENTITY oacute SDATA "[oacute]"--=small o, acute accent-->
<!ENTITY Oacute SDATA "[Oacute]"--=capital O, acute accent-->
<!ENTITY ocirc  SDATA "[ocirc ]"--=small o, circumflex accent-->
```

```
<!ENTITY Ocirc  SDATA "[Ocirc ]"--=capital O, circumflex accent-->
<!ENTITY ograve SDATA "[ograve]"--=small o, grave accent-->
<!ENTITY Ograve SDATA "[Ograve]"--=capital O, grave accent-->
<!ENTITY oslash SDATA "[oslash]"--=small o, slash-->
<!ENTITY Oslash SDATA "[Oslash]"--=capital O, slash-->
<!ENTITY otilde SDATA "[otilde]"--=small o, tilde-->
<!ENTITY Otilde SDATA "[Otilde]"--=capital O, tilde-->
<!ENTITY ouml   SDATA "[ouml  ]"--=small o, dieresis or umlaut mark-->
<!ENTITY Ouml   SDATA "[Ouml  ]"--=capital O, dieresis or umlaut mark-->
<!ENTITY szlig  SDATA "[szlig ]"--=small sharp s, German (sz lig.)-->
<!ENTITY thorn  SDATA "[thorn ]"--=small thorn, Icelandic-->
<!ENTITY THORN  SDATA "[THORN ]"--=capital THORN, Icelandic-->
<!ENTITY uacute SDATA "[uacute]"--=small u, acute accent-->
<!ENTITY Uacute SDATA "[Uacute]"--=capital U, acute accent-->
<!ENTITY ucirc  SDATA "[ucirc ]"--=small u, circumflex accent-->
<!ENTITY Ucirc  SDATA "[Ucirc ]"--=capital U, circumflex accent-->
<!ENTITY ugrave SDATA "[ugrave]"--=small u, grave accent-->
<!ENTITY Ugrave SDATA "[Ugrave]"--=capital U, grave accent-->
<!ENTITY uuml   SDATA "[uuml  ]"--=small u, dieresis or umlaut mark-->
<!ENTITY Uuml   SDATA "[Uuml  ]"--=capital U, dieresis or umlaut mark-->
<!ENTITY yacute SDATA "[yacute]"--=small y, acute accent-->
<!ENTITY Yacute SDATA "[Yacute]"--=capital Y, acute accent-->
<!ENTITY yuml   SDATA "[yuml  ]"--=small y, dieresis or umlaut mark-->
```

%ISOlat2; I.20

An entity reference to a set of ISO standard character entity references for accented Latin characters used in eastern European languages.

```
<!ENTITY % ISOlat2 PUBLIC "ISO 8879-1986//ENTITIES Added Latin 2//EN">
%ISOlat2;
```

This is the entity set:

```
<!-- (C) International Organization for Standardization 1986
     Permission to copy in any form is granted for use with
     conforming SGML systems and applications as defined in
     ISO 8879, provided this notice is included in all copies.-->
<!ENTITY abreve SDATA "[abreve]"--=small a, breve-->
<!ENTITY Abreve SDATA "[Abreve]"--=capital A, breve-->
<!ENTITY amacr  SDATA "[amacr ]"--=small a, macron-->
<!ENTITY Amacr  SDATA "[Amacr ]"--=capital A, macron-->
<!ENTITY aogon  SDATA "[aogon ]"--=small a, ogonek-->
<!ENTITY Aogon  SDATA "[Aogon ]"--=capital A, ogonek-->
<!ENTITY cacute SDATA "[cacute]"--=small c, acute accent-->
<!ENTITY Cacute SDATA "[Cacute]"--=capital C, acute accent-->
<!ENTITY ccaron SDATA "[ccaron]"--=small c, caron-->
<!ENTITY Ccaron SDATA "[Ccaron]"--=capital C, caron-->
<!ENTITY ccirc  SDATA "[ccirc ]"--=small c, circumflex accent-->
<!ENTITY Ccirc  SDATA "[Ccirc ]"--=capital C, circumflex accent-->
<!ENTITY cdot   SDATA "[cdot  ]"--=small c, dot above-->
<!ENTITY Cdot   SDATA "[Cdot  ]"--=capital C, dot above-->
<!ENTITY dcaron SDATA "[dcaron]"--=small d, caron-->
<!ENTITY Dcaron SDATA "[Dcaron]"--=capital D, caron-->
<!ENTITY dstrok SDATA "[dstrok]"--=small d, stroke-->
```

```
<!ENTITY Dstrok SDATA "[Dstrok]"--=capital D, stroke-->
<!ENTITY ecaron SDATA "[ecaron]"--=small e, caron-->
<!ENTITY Ecaron SDATA "[Ecaron]"--=capital E, caron-->
<!ENTITY edot   SDATA "[edot  ]"--=small e, dot above-->
<!ENTITY Edot   SDATA "[Edot  ]"--=capital E, dot above-->
<!ENTITY emacr  SDATA "[emacr ]"--=small e, macron-->
<!ENTITY Emacr  SDATA "[Emacr ]"--=capital E, macron-->
<!ENTITY eogon  SDATA "[eogon ]"--=small e, ogonek-->
<!ENTITY Eogon  SDATA "[Eogon ]"--=capital E, ogonek-->
<!ENTITY gacute SDATA "[gacute]"--=small g, acute accent-->
<!ENTITY gbreve SDATA "[gbreve]"--=small g, breve-->
<!ENTITY Gbreve SDATA "[Gbreve]"--=capital G, breve-->
<!ENTITY Gcedil SDATA "[Gcedil]"--=capital G, cedilla-->
<!ENTITY gcirc  SDATA "[gcirc ]"--=small g, circumflex accent-->
<!ENTITY Gcirc  SDATA "[Gcirc ]"--=capital G, circumflex accent-->
<!ENTITY gdot   SDATA "[gdot  ]"--=small g, dot above-->
<!ENTITY Gdot   SDATA "[Gdot  ]"--=capital G, dot above-->
<!ENTITY hcirc  SDATA "[hcirc ]"--=small h, circumflex accent-->
<!ENTITY Hcirc  SDATA "[Hcirc ]"--=capital H, circumflex accent-->
<!ENTITY hstrok SDATA "[hstrok]"--=small h, stroke-->
<!ENTITY Hstrok SDATA "[Hstrok]"--=capital H, stroke-->
<!ENTITY Idot   SDATA "[Idot  ]"--=capital I, dot above-->
<!ENTITY Imacr  SDATA "[Imacr ]"--=capital I, macron-->
<!ENTITY imacr  SDATA "[imacr ]"--=small i, macron-->
<!ENTITY ijlig  SDATA "[ijlig ]"--=small ij ligature-->
<!ENTITY IJlig  SDATA "[IJlig ]"--=capital IJ ligature-->
<!ENTITY inodot SDATA "[inodot]"--=small i without dot-->
<!ENTITY iogon  SDATA "[iogon ]"--=small i, ogonek-->
<!ENTITY Iogon  SDATA "[Iogon ]"--=capital I, ogonek-->
<!ENTITY itilde SDATA "[itilde]"--=small i, tilde-->
<!ENTITY Itilde SDATA "[Itilde]"--=capital I, tilde-->
<!ENTITY jcirc  SDATA "[jcirc ]"--=small j, circumflex accent-->
<!ENTITY Jcirc  SDATA "[Jcirc ]"--=capital J, circumflex accent-->
<!ENTITY kcedil SDATA "[kcedil]"--=small k, cedilla-->
<!ENTITY Kcedil SDATA "[Kcedil]"--=capital K, cedilla-->
<!ENTITY kgreen SDATA "[kgreen]"--=small k, Greenlandic-->
<!ENTITY lacute SDATA "[lacute]"--=small l, acute accent-->
<!ENTITY Lacute SDATA "[Lacute]"--=capital L, acute accent-->
<!ENTITY lcaron SDATA "[lcaron]"--=small l, caron-->
<!ENTITY Lcaron SDATA "[Lcaron]"--=capital L, caron-->
<!ENTITY lcedil SDATA "[lcedil]"--=small l, cedilla-->
<!ENTITY Lcedil SDATA "[Lcedil]"--=capital L, cedilla-->
<!ENTITY lmidot SDATA "[lmidot]"--=small l, middle dot-->
<!ENTITY Lmidot SDATA "[Lmidot]"--=capital L, middle dot-->
<!ENTITY lstrok SDATA "[lstrok]"--=small l, stroke-->
<!ENTITY Lstrok SDATA "[Lstrok]"--=capital L, stroke-->
<!ENTITY nacute SDATA "[nacute]"--=small n, acute accent-->
<!ENTITY Nacute SDATA "[Nacute]"--=capital N, acute accent-->
<!ENTITY eng    SDATA "[eng   ]"--=small eng, Lapp-->
<!ENTITY ENG    SDATA "[ENG   ]"--=capital ENG, Lapp-->
<!ENTITY napos  SDATA "[napos ]"--=small n, apostrophe-->
<!ENTITY ncaron SDATA "[ncaron]"--=small n, caron-->
<!ENTITY Ncaron SDATA "[Ncaron]"--=capital N, caron-->
<!ENTITY ncedil SDATA "[ncedil]"--=small n, cedilla-->
<!ENTITY Ncedil SDATA "[Ncedil]"--=capital N, cedilla-->
```

```
<!ENTITY odblac SDATA "[odblac]"--=small o, double acute accent-->
<!ENTITY Odblac SDATA "[Odblac]"--=capital O, double acute accent-->
<!ENTITY Omacr  SDATA "[Omacr ]"--=capital O, macron-->
<!ENTITY omacr  SDATA "[omacr ]"--=small o, macron-->
<!ENTITY oelig  SDATA "[oelig ]"--=small oe ligature-->
<!ENTITY OElig  SDATA "[OElig ]"--=capital OE ligature-->
<!ENTITY racute SDATA "[racute]"--=small r, acute accent-->
<!ENTITY Racute SDATA "[Racute]"--=capital R, acute accent-->
<!ENTITY rcaron SDATA "[rcaron]"--=small r, caron-->
<!ENTITY Rcaron SDATA "[Rcaron]"--=capital R, caron-->
<!ENTITY rcedil SDATA "[rcedil]"--=small r, cedilla-->
<!ENTITY Rcedil SDATA "[Rcedil]"--=capital R, cedilla-->
<!ENTITY sacute SDATA "[sacute]"--=small s, acute accent-->
<!ENTITY Sacute SDATA "[Sacute]"--=capital S, acute accent-->
<!ENTITY scaron SDATA "[scaron]"--=small s, caron-->
<!ENTITY Scaron SDATA "[Scaron]"--=capital S, caron-->
<!ENTITY scedil SDATA "[scedil]"--=small s, cedilla-->
<!ENTITY Scedil SDATA "[Scedil]"--=capital S, cedilla-->
<!ENTITY scirc  SDATA "[scirc ]"--=small s, circumflex accent-->
<!ENTITY Scirc  SDATA "[Scirc ]"--=capital S, circumflex accent-->
<!ENTITY tcaron SDATA "[tcaron]"--=small t, caron-->
<!ENTITY Tcaron SDATA "[Tcaron]"--=capital T, caron-->
<!ENTITY tcedil SDATA "[tcedil]"--=small t, cedilla-->
<!ENTITY Tcedil SDATA "[Tcedil]"--=capital T, cedilla-->
<!ENTITY tstrok SDATA "[tstrok]"--=small t, stroke-->
<!ENTITY Tstrok SDATA "[Tstrok]"--=capital T, stroke-->
<!ENTITY ubreve SDATA "[ubreve]"--=small u, breve-->
<!ENTITY Ubreve SDATA "[Ubreve]"--=capital U, breve-->
<!ENTITY udblac SDATA "[udblac]"--=small u, double acute accent-->
<!ENTITY Udblac SDATA "[Udblac]"--=capital U, double acute accent-->
<!ENTITY umacr  SDATA "[umacr ]"--=small u, macron-->
<!ENTITY Umacr  SDATA "[Umacr ]"--=capital U, macron-->
<!ENTITY uogon  SDATA "[uogon ]"--=small u, ogonek-->
<!ENTITY Uogon  SDATA "[Uogon ]"--=capital U, ogonek-->
<!ENTITY uring  SDATA "[uring ]"--=small u, ring-->
<!ENTITY Uring  SDATA "[Uring ]"--=capital U, ring-->
<!ENTITY utilde SDATA "[utilde]"--=small u, tilde-->
<!ENTITY Utilde SDATA "[Utilde]"--=capital U, tilde-->
<!ENTITY wcirc  SDATA "[wcirc ]"--=small w, circumflex accent-->
<!ENTITY Wcirc  SDATA "[Wcirc ]"--=capital W, circumflex accent-->
<!ENTITY ycirc  SDATA "[ycirc ]"--=small y, circumflex accent-->
<!ENTITY Ycirc  SDATA "[Ycirc ]"--=capital Y, circumflex accent-->
<!ENTITY Yuml   SDATA "[Yuml  ]"--=capital Y, dieresis or umlaut mark-->
<!ENTITY zacute SDATA "[zacute]"--=small z, acute accent-->
<!ENTITY Zacute SDATA "[Zacute]"--=capital Z, acute accent-->
<!ENTITY zcaron SDATA "[zcaron]"--=small z, caron-->
<!ENTITY Zcaron SDATA "[Zcaron]"--=capital Z, caron-->
<!ENTITY zdot   SDATA "[zdot  ]"--=small z, dot above-->
<!ENTITY Zdot   SDATA "[Zdot  ]"--=capital Z, dot above-->
```

%ISOnum; I.21

An entity reference to a set of ISO standard character entity references for numeric and special characters, most of which are in the Symbol font.

```
<!ENTITY % ISOnum PUBLIC "ISO 8879-1986//ENTITIES
          Numeric and Special Graphic//EN">
%ISOnum;
```

This is the entity set:

```
<!-- (C) International Organization for Standardization 1986
     Permission to copy in any form is granted for use with
     conforming SGML systems and applications as defined in
     ISO 8879, provided this notice is included in all copies.-->
<!ENTITY half   SDATA "[half  ]"--=fraction one-half-->
<!ENTITY frac12 SDATA "[frac12]"--=fraction one-half-->
<!ENTITY frac14 SDATA "[frac14]"--=fraction one-quarter-->
<!ENTITY frac34 SDATA "[frac34]"--=fraction three-quarters-->
<!ENTITY frac18 SDATA "[frac18]"--=fraction one-eighth-->
<!ENTITY frac38 SDATA "[frac38]"--=fraction three-eighths-->
<!ENTITY frac58 SDATA "[frac58]"--=fraction five-eighths-->
<!ENTITY frac78 SDATA "[frac78]"--=fraction seven-eighths-->
<!ENTITY sup1   SDATA "[sup1  ]"--=superscript one-->
<!ENTITY sup2   SDATA "[sup2  ]"--=superscript two-->
<!ENTITY sup3   SDATA "[sup3  ]"--=superscript three-->
<!ENTITY plus   SDATA "[plus  ]"--=plus sign B:-- >
<!ENTITY plusmn SDATA "[plusmn]"--/pm B: =plus-or-minus sign-->
<!ENTITY lt     SDATA "[lt    ]"--=less-than sign R:-->
<!ENTITY equals SDATA "[equals]"--=equals sign R:-->
<!ENTITY gt     SDATA "[gt    ]"--=greater-than sign R:-->
<!ENTITY divide SDATA "[divide]"--/div B: =divide sign-->
<!ENTITY times  SDATA "[times ]"--/times B: =multiply sign-->
<!ENTITY curren SDATA "[curren]"--=general currency sign-->
<!ENTITY pound  SDATA "[pound ]"--=pound sign-->
<!ENTITY dollar SDATA "[dollar]"--=dollar sign-->
<!ENTITY cent   SDATA "[cent  ]"--=cent sign-->
<!ENTITY yen    SDATA "[yen   ]"--/yen =yen sign-->
<!ENTITY num    SDATA "[num   ]"--=number sign-->
<!ENTITY percnt SDATA "[percnt]"--=percent sign-->
<!ENTITY amp    SDATA "[amp   ]"--=ampersand-->
<!ENTITY ast    SDATA "[ast   ]"--/ast B: =asterisk-->
<!ENTITY commat SDATA "[commat]"--=commercial at-->
<!ENTITY lsqb   SDATA "[lsqb  ]"--/lbrack O: =left square bracket-->
<!ENTITY bsol   SDATA "[bsol  ]"--/backslash =reverse solidus-->
<!ENTITY rsqb   SDATA "[rsqb  ]"--/rbrack C: =right square bracket-->
<!ENTITY lcub   SDATA "[lcub  ]"--/lbrace O: =left curly bracket-->
<!ENTITY horbar SDATA "[horbar]"--=horizontal bar-->
<!ENTITY verbar SDATA "[verbar]"--/vert =vertical bar-->
<!ENTITY rcub   SDATA "[rcub  ]"--/rbrace C: =right curly bracket-->
<!ENTITY micro  SDATA "[micro ]"--=micro sign-->
<!ENTITY ohm    SDATA "[ohm   ]"--=ohm sign-->
<!ENTITY deg    SDATA "[deg   ]"--=degree sign-->
<!ENTITY ordm   SDATA "[ordm  ]"--=ordinal indicator, masculine-->
<!ENTITY ordf   SDATA "[ordf  ]"--=ordinal indicator, feminine-->
<!ENTITY sect   SDATA "[sect  ]"--=section sign-->
<!ENTITY para   SDATA "[para  ]"--=pilcrow (paragraph sign)-->
<!ENTITY middot SDATA "[middot]"--/centerdot B: =middle dot-->
<!ENTITY larr   SDATA "[larr  ]"--/leftarrow /gets A: =leftward arrow-->
<!ENTITY rarr   SDATA "[rarr  ]"--/rightarrow /to A: =rightward arrow-->
<!ENTITY uarr   SDATA "[uarr  ]"--/uparrow A: =upward arrow-->
```

```
<!ENTITY darr   SDATA "[darr  ]"--/downarrow A: =downward arrow-->
<!ENTITY copy   SDATA "[copy  ]"--=copyright sign-->
<!ENTITY reg    SDATA "[reg   ]"--/circledR =registered sign-->
<!ENTITY trade  SDATA "[trade ]"--=trade mark sign-->
<!ENTITY brvbar SDATA "[brvbar]"--=broken (vertical) bar-->
<!ENTITY not    SDATA "[not   ]"--/neg /lnot =not sign-->
<!ENTITY sung   SDATA "[sung  ]"--=music note (sung text sign)-->
<!ENTITY excl   SDATA "[excl  ]"--=exclamation mark-->
<!ENTITY iexcl  SDATA "[iexcl ]"--=inverted exclamation mark-->
<!ENTITY quot   SDATA "[quot  ]"--=quotation mark-->
<!ENTITY apos   SDATA "[apos  ]"--=apostrophe-->
<!ENTITY lpar   SDATA "[lpar  ]"--O: =left parenthesis-->
<!ENTITY rpar   SDATA "[rpar  ]"--C: =right parenthesis-->
<!ENTITY comma  SDATA "[comma ]"--P: =comma-->
<!ENTITY lowbar SDATA "[lowbar]"--=low line-->
<!ENTITY hyphen SDATA "[hyphen]"--=hyphen-->
<!ENTITY period SDATA "[period]"--=full stop, period-->
<!ENTITY sol    SDATA "[sol   ]"--=solidus-->
<!ENTITY colon  SDATA "[colon ]"--/colon P:-->
<!ENTITY semi   SDATA "[semi  ]"--=semicolon P:-->
<!ENTITY quest  SDATA "[quest ]"--=question mark-->
<!ENTITY iquest SDATA "[iquest]"--=inverted question mark-->
<!ENTITY laquo  SDATA "[laquo ]"--=angle quotation mark, left-->
<!ENTITY raquo  SDATA "[raquo ]"--=angle quotation mark, right-->
<!ENTITY lsquo  SDATA "[lsquo ]"--=single quotation mark, left-->
<!ENTITY rsquo  SDATA "[rsquo ]"--=single quotation mark, right-->
<!ENTITY ldquo  SDATA "[ldquo ]"--=double quotation mark, left-->
<!ENTITY rdquo  SDATA "[rdquo ]"--=double quotation mark, right-->
<!ENTITY nbsp   SDATA "[nbsp  ]"--=no break (required) space-->
<!ENTITY shy    SDATA "[shy   ]"--=soft hyphen-->
```

%ISOpub; I.22

An entity reference to a set of ISO standard character entity references for publishers' special characters, some of which are in the Symbol or IBMPCDOS font.

```
<!ENTITY % ISOpub PUBLIC "ISO 8879-1986//ENTITIES Publishing//EN">
%ISOpub;
```

This is the entity set:

```
<!-- (C) International Organization for Standardization 1986
     Permission to copy in any form is granted for use with
     conforming SGML systems and applications as defined in
     ISO 8879, provided this notice is included in all copies.-->
<!ENTITY emsp   SDATA "[emsp  ]"--=em space-->
<!ENTITY ensp   SDATA "[ensp  ]"--=en space (1/2-em)-->
<!ENTITY emsp13 SDATA "[emsp3 ]"--=1/3-em space-->
<!ENTITY emsp14 SDATA "[emsp4 ]"--=1/4-em space-->
<!ENTITY numsp  SDATA "[numsp ]"--=digit space (width of a number)-->
<!ENTITY puncsp SDATA "[puncsp]"--=punctuation space (width of comma)-->
<!ENTITY thinsp SDATA "[thinsp]"--=thin space (1/6-em)-->
<!ENTITY hairsp SDATA "[hairsp]"--=hair space-->
<!ENTITY mdash  SDATA "[mdash ]"--=em dash-->
```

```
<!ENTITY ndash  SDATA "[ndash ]"--=en dash-->
<!ENTITY dash   SDATA "[dash  ]"--=hyphen (true graphic)-->
<!ENTITY blank  SDATA "[blank ]"--=significant blank symbol-->
<!ENTITY hellip SDATA "[hellip]"--=ellipsis (horizontal)-->
<!ENTITY nldr   SDATA "[nldr  ]"--=double baseline dot (en leader)-->
<!ENTITY frac13 SDATA "[frac13]"--=fraction one-third-->
<!ENTITY frac23 SDATA "[frac23]"--=fraction two-thirds-->
<!ENTITY frac15 SDATA "[frac15]"--=fraction one-fifth-->
<!ENTITY frac25 SDATA "[frac25]"--=fraction two-fifths-->
<!ENTITY frac35 SDATA "[frac35]"--=fraction three-fifths-->
<!ENTITY frac45 SDATA "[frac45]"--=fraction four-fifths-->
<!ENTITY frac16 SDATA "[frac16]"--=fraction one-sixth-->
<!ENTITY frac56 SDATA "[frac56]"--=fraction five-sixths-->
<!ENTITY incare SDATA "[incare]"--=in-care-of symbol-->
<!ENTITY block  SDATA "[block ]"--=full block-->
<!ENTITY uhblk  SDATA "[uhblk ]"--=upper half block-->
<!ENTITY lhblk  SDATA "[lhblk ]"--=lower half block-->
<!ENTITY blk14  SDATA "[blk14 ]"--=25% shaded block-->
<!ENTITY blk12  SDATA "[blk12 ]"--=50% shaded block-->
<!ENTITY blk34  SDATA "[blk34 ]"--=75% shaded block-->
<!ENTITY marker SDATA "[marker]"--=histogram marker-->
<!ENTITY cir    SDATA "[cir   ]"--/circ B: =circle, open-->
<!ENTITY squ    SDATA "[squ   ]"--=square, open-->
<!ENTITY rect   SDATA "[rect  ]"--=rectangle, open-->
<!ENTITY utri   SDATA "[utri  ]"--/triangle =up triangle, open-->
<!ENTITY dtri   SDATA "[dtri  ]"--/triangledown =down triangle, open-->
<!ENTITY star   SDATA "[star  ]"--=star, open-->
<!ENTITY bull   SDATA "[bull  ]"--/bullet B: =round bullet, filled-->
<!ENTITY squf   SDATA "[squf  ]"--/blacksquare =sq bullet, filled-->
<!ENTITY utrif  SDATA "[utrif ]"--/blacktriangle =up tri, filled-->
<!ENTITY dtrif  SDATA "[dtrif ]"--/blacktriangledown =dn tri, filled-->
<!ENTITY ltrif  SDATA "[ltrif ]"--/blacktriangleleft R: =l tri,filled-->
<!ENTITY rtrif  SDATA "[rtrif ]"--/blacktriangleright R:=r tri,filled-->
<!ENTITY clubs  SDATA "[clubs ]"--/clubsuit =club suit symbol-->
<!ENTITY diams  SDATA "[diams ]"--/diamondsuit =diamond suit symbol-->
<!ENTITY hearts SDATA "[hearts]"--/heartsuit =heart suit symbol-->
<!ENTITY spades SDATA "[spades]"--/spadesuit =spades suit symbol-->
<!ENTITY malt   SDATA "[malt  ]"--/maltese =maltese cross-->
<!ENTITY dagger SDATA "[dagger]"--/dagger B: =dagger-->
<!ENTITY Dagger SDATA "[Dagger]"--/ddagger B: =double dagger-->
<!ENTITY check  SDATA "[check ]"--/checkmark =tick, check mark-->
<!ENTITY cross  SDATA "[ballot]"--=ballot cross-->
<!ENTITY sharp  SDATA "[sharp ]"--/sharp =musical sharp-->
<!ENTITY flat   SDATA "[flat  ]"--/flat =musical flat-->
<!ENTITY male   SDATA "[male  ]"--=male symbol-->
<!ENTITY female SDATA "[female]"--=female symbol-->
<!ENTITY phone  SDATA "[phone ]"--=telephone symbol-->
<!ENTITY telrec SDATA "[telrec]"--=telephone recorder symbol-->
<!ENTITY copysr SDATA "[copysr]"--=sound recording copyright sign-->
<!ENTITY caret  SDATA "[caret ]"--=caret (insertion mark)-->
<!ENTITY lsquor SDATA "[lsquor]"--=rising single quote, left (low)-->
<!ENTITY ldquor SDATA "[ldquor]"--=rising dbl quote, left (low)-->
<!ENTITY fflig  SDATA "[fflig ]"--small ff ligature-->
<!ENTITY filig  SDATA "[filig ]"--small fi ligature-->
<!ENTITY fjlig  SDATA "[fjlig ]"--small fj ligature-->
```

```
<!ENTITY ffilig SDATA "[ffilig]"--small ffi ligature-->
<!ENTITY ffllig SDATA "[ffllig]"--small ffl ligature-->
<!ENTITY fllig  SDATA "[fllig ]"--small fl ligature-->
<!ENTITY mldr   SDATA "[mldr  ]"--em leader-->
<!ENTITY rdquor SDATA "[rdquor]"--rising dbl quote, right (high)-->
<!ENTITY rsquor SDATA "[rsquor]"--rising single quote, right (high)-->
<!ENTITY vellip SDATA "[vellip]"--vertical ellipsis-->
<!ENTITY hybull SDATA "[hybull]"--rectangle, filled (hyphen bullet)-->
<!ENTITY loz    SDATA "[loz   ]"--/lozenge - lozenge or total mark-->
<!ENTITY lozf   SDATA "[lozf  ]"--/blacklozenge - lozenge, filled-->
<!ENTITY ltri   SDATA "[ltri  ]"--/triangleleft B: l triangle, open-->
<!ENTITY rtri   SDATA "[rtri  ]"--/triangleright B: r triangle, open-->
<!ENTITY starf  SDATA "[starf ]"--/bigstar - star, filled-->
<!ENTITY natur  SDATA "[natur ]"--/natural - music natural-->
<!ENTITY rx     SDATA "[rx    ]"--pharmaceutical prescription (Rx)-->
<!ENTITY sext   SDATA "[sext  ]"--sextile (6-pointed star)-->
<!ENTITY target SDATA "[target]"--register mark or target-->
<!ENTITY dlcrop SDATA "[dlcrop]"--downward left crop mark -->
<!ENTITY drcrop SDATA "[drcrop]"--downward right crop mark -->
<!ENTITY ulcrop SDATA "[ulcrop]"--upward left crop mark -->
<!ENTITY urcrop SDATA "[urcrop]"--upward right crop mark -->
```

%ISOtech; I.23

An entity reference to a set of ISO standard character entity references for technical characters, some of which are in the Symbol font.

```
<!ENTITY % ISOtech PUBLIC "ISO 8879-1986//ENTITIES General Technical//
EN">
%ISOtech;
```

This is the entity set:

```
<!-- (C) International Organization for Standardization 1986
     Permission to copy in any form is granted for use with
     conforming SGML systems and applications as defined in
     ISO 8879, provided this notice is included in all copies.-->
<!ENTITY aleph  SDATA "[aleph ]"--/aleph =aleph, Hebrew-->
<!ENTITY and    SDATA "[and   ]"--/wedge /land B: =logical and-->
<!ENTITY ang90  SDATA "[ang90 ]"--=right (90 degree) angle-->
<!ENTITY angsph SDATA "[angsph]"--/sphericalangle =angle-spherical-->
<!ENTITY ap     SDATA "[ap    ]"--/approx R: =approximate-->
<!ENTITY becaus SDATA "[becaus]"--/because R: =because-->
<!ENTITY bottom SDATA "[bottom]"--/bot B: =perpendicular-->
<!ENTITY cap    SDATA "[cap   ]"--/cap B: =intersection-->
<!ENTITY cong   SDATA "[cong  ]"--/cong R: =congruent with-->
<!ENTITY conint SDATA "[conint]"--/oint L: =contour integral operator-->
<!ENTITY cup    SDATA "[cup   ]"--/cup B: =union or logical sum-->
<!ENTITY equiv  SDATA "[equiv ]"--/equiv R: =identical with-->
<!ENTITY exist  SDATA "[exist ]"--/exists =at least one exists-->
<!ENTITY forall SDATA "[forall]"--/forall =for all-->
<!ENTITY fnof   SDATA "[fnof  ]"--=function of (italic small f)-->
<!ENTITY ge     SDATA "[ge    ]"--/geq /ge R: =greater-than-or-equal-->
<!ENTITY iff    SDATA "[iff   ]"--/iff =if and only if-->
<!ENTITY infin  SDATA "[infin ]"--/infty =infinity-->
```

```
<!ENTITY int    SDATA "[int   ]"--/int L: =integral operator-->
<!ENTITY isin   SDATA "[isin  ]"--/in R: =set membership-->
<!ENTITY lang   SDATA "[lang  ]"--/langle O: =left angle bracket-->
<!ENTITY lArr   SDATA "[lArr  ]"--/Leftarrow A: =is implied by-->
<!ENTITY le     SDATA "[le    ]"--/leq /le R: =less-than-or-equal-->
<!ENTITY minus  SDATA "[minus ]"--B: =minus sign-->
<!ENTITY mnplus SDATA "[mnplus]"--/mp B: =minus-or-plus sign-->
<!ENTITY nabla  SDATA "[nabla ]"--/nabla =del, Hamilton operator-->
<!ENTITY ne     SDATA "[ne    ]"--/ne /neq R: =not equal-->
<!ENTITY ni     SDATA "[ni    ]"--/ni /owns R: =contains-->
<!ENTITY or     SDATA "[or    ]"--/vee /lor B: =logical or-->
<!ENTITY par    SDATA "[par   ]"--/parallel R: =parallel-->
<!ENTITY part   SDATA "[part  ]"--/partial =partial differential-->
<!ENTITY permil SDATA "[permil]"--=per thousand-->
<!ENTITY perp   SDATA "[perp  ]"--/perp R: =perpendicular-->
<!ENTITY prime  SDATA "[prime ]"--/prime =prime or minute-->
<!ENTITY Prime  SDATA "[Prime ]"--=double prime or second-->
<!ENTITY prop   SDATA "[prop  ]"--/propto R: =is proportional to-->
<!ENTITY radic  SDATA "[radic ]"--/surd =radical-->
<!ENTITY rang   SDATA "[rang  ]"--/rangle C: =right angle bracket-->
<!ENTITY rArr   SDATA "[rArr  ]"--/Rightarrow A: =implies-->
<!ENTITY sim    SDATA "[sim   ]"--/sim R: =similar-->
<!ENTITY sime   SDATA "[sime  ]"--/simeq R: =similar, equals-->
<!ENTITY square SDATA "[square]"--/square B: =square-->
<!ENTITY sub    SDATA "[sub   ]"--/subset R: =subset or is implied by-->
<!ENTITY sube   SDATA "[sube  ]"--/subseteq R: =subset, equals-->
<!ENTITY sup    SDATA "[sup   ]"--/supset R: =superset or implies-->
<!ENTITY supe   SDATA "[supe  ]"--/supseteq R: =superset, equals-->
<!ENTITY there4 SDATA "[there4]"--/therefore R: =therefore-->
<!ENTITY Verbar SDATA "[Verbar]"--/Vert =dbl vertical bar-->
<!ENTITY angst  SDATA "[angst ]"--Angstrom =capital A, ring-->
<!ENTITY bernou SDATA "[bernou]"--Bernoulli function (script cap B)-->
<!ENTITY compfn SDATA "[compfn]"--B:composite function (small circle)-->
<!ENTITY Dot    SDATA "[Dot   ]"--=dieresis or umlaut mark-->
<!ENTITY DotDot SDATA "[DotDot]"--four dots above-->
<!ENTITY hamilt SDATA "[hamilt]"--Hamiltonian (script capital H)-->
<!ENTITY lagran SDATA "[lagran]"--Lagrangian (script capital L)-->
<!ENTITY lowast SDATA "[lowast]"--low asterisk-->
<!ENTITY notin  SDATA "[notin ]"--N: negated set membership-->
<!ENTITY order  SDATA "[order ]"--order of (script small o)-->
<!ENTITY phmmat SDATA "[phmmat]"--physics M-matrix (script capital M)-->
<!ENTITY tdot   SDATA "[tdot  ]"--three dots above-->
<!ENTITY tprime SDATA "[tprime]"--triple prime-->
<!ENTITY wedgeq SDATA "[wedgeq]"--R: corresponds to (wedge, equals)-->
```

"Issue", type= — DistOrdLI (D.6), T (T.1)

"ital", emph= (E.6)

J (JFIF)

JFIF (notation) J.1

Notation that should be declared for a `filename.jpg` graphic (Joint Photographic Experts Group File Interchange Format).

```
<!NOTATION JFIF      PUBLIC "+//ISBN 0-7923-9432-1::Graphic
          Notation//NOTATION JPEG File Interchange Format//EN">
```

Example:

```
<!ENTITY fig001 SYSTEM "../jpeg/fig001.jpg" NDATA JFIF>
```

The file should be stored in a `jpeg` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s). Currently the TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG (P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18).

JPEG files should be JFIF version 1.x, which is read by more applications than 2.x versions. The version number is encoded in byte positions 11 and 12 of the file (counting the first byte as position 0), after "JFIF" and a 00h character. If the next character is 01h, the file is version 1.x; if it is 02h, the file is version 2.x. JPEG files are 24-bit (up to 16.7 million colors), but they smooth out color variations to achieve good file compression; that is, compression is "lossy," but in ways that are designed to be hard to notice. Even with imperceptible loss of detail, photo-image files can be much smaller than in any other file format, at least 75% smaller than uncompressed files. If some noticeable loss of detail is acceptable, compression can be up to 98%. But JPEG compression is not good choice for line drawings that use just a few distinct colors; it can produce "ghosts" near sharp edges. Any other bitmap format is preferable for line drawings, with TIFFGRP4 offering the most compression.

JPEG — JFIF (J.1)

K (Keywords — keywords)

<Keycaps> (TIM 1 only)

Replaced in TIM 2 by a more general element, **Symbol** (S.33).

<Keywords> K.1

May occur once within **DocID** (D.27) to provide a list of keywords for the document (**keywords** is also an attribute, K.2).

```
<!ELEMENT Keywords    - -  (#PCDATA)* >
<!ATTLIST Keywords
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Specific forms for the information it contains are left to the discretion of the document originator. There is no length limit, but also no formatting — tabs and line breaks will be treated as spaces. Commas or semicolons should be used to separate distinct key terms, which might be multiple words. Example:

```
<Keywords>document numbers, document identifiers, information
product identifiers, numbering, document control coordinator</
Keywords>
```

keywords= K.2

Keywords — any that the author considers useful for cross-referencing or categorizing an element (**Keywords** is also an element, K.1).

```
keywords CDATA #IMPLIED
```

The default is null — no keywords. Use a comma (,) or semicolon (;) between separate keywords:

```
<Section type="Chapter" keywords="element, attribute, TIM">
```

This attribute is usually defined through the entity reference **%identity;** (I.3).

Comment

Keyword values can be used to provide anything a browser could search for that will aid in locating documents, sections, passages, terms, and, most likely, graphics, whose own text content will not be found in a search. Normally the keywords will not be visible when reading the text, but provide additional search targets besides the words that occur in the text content. However, authors are encouraged to use keywords thoroughly and systematically, on the assumption that they will be the most reliable targets for searching, in which case words in the text content will often be repeated as keywords in the attribute value.

L (label — locsrc)

label= L.1

Contains an element's number or mark (e.g., "Section 1.2" or "•") if it is not left to be calculated by the rendering application. Using the label supplied by the author is a way around complex algorithms for numbering of items in documents when the rendering application is unsophisticated. Labels may also be needed to resolve manually applied (as opposed to system-generated) cross-references — which, of course, are to be strongly discouraged.

For most elements, a label would normally match the specifications of its numeration attributes. But a label on a list item would normally match the specifications of its parent list or list group's numeration or mark attributes.

Parsers cannot enforce consistency between the **`label`** value and the type of **`numeration`** or **`mark`** specified, so ideally there will be processing rules in the rendering application to handle inconsistencies like "**`<Section numeration='UpperAlpha' label='2.'>`**."

```
label CDATA #IMPLIED
```

Default value: undefined, meaning that the label should be derived from numbering attributes (begin by checking **`numeration`** or **`mark`**) or applied algorithmically from the recipient's own rules (e.g., that all first-level unordered lists should be bulleted). Since some applications can't distinguish a null attribute value, "", from an undefined one, the special value "nolabel" serves as an explicit "", meaning use no label rather than choosing a different default.

This attribute is usually defined through the entity reference **`%identity;`** (I.3).

Comment

Labels should always be supplied for enumerated elements, because rendering applications can't be expected to understand all possible autonumbering algorithms. Document providers should assume that users will rely on the value of **`label`** for indexing and cross-referencing whenever possible, rather than try to regenerate numbers automatically. The various numeration attributes (see N.13), which specify the algorithm used, will generally be of interest only to those who are going to revise the document or reuse its content.

Style Notes

Although the label is defined as CDATA notation, entity references like **`&bull;`** (bullet) and **`&mdash;`** (em dash) should be replaced by the referenced characters during processing (as if the notation were RCDATA); this is inconsistent but correct behavior according to the SGML standard. However, it is a heavy burden on rendering applications to find and replace character entity references anywhere in an attribute value. Therefore it is strongly recommended that

label contain either a single entity reference and no other characters or else a string that does not contain an entity reference.

The value specified for **label** should not include a separator: **label="2."**, not **label="2. "**. The separator should be supplied by the rendering application.

"Label", type= — T (T.1)

<Lang> L.2

An optional element of **DocID** (D.27) used to identify the language of the document (**lang** is also an attribute, L.3).

```
<!ELEMENT Lang    - -  (#PCDATA)* >
<!ATTLIST Lang
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

If a language code is used to distinguish separately orderable versions of a document, that code should appear in a **DN.lang** (D.17) subelement of **DocNo**, whether or not it also appears in **Lang**.

Example:

```
<Lang>EN</Lang>
```

Use of the ISO-639 codes for languages (Table 12-1) is encouraged but not required here (they are required in the **lang** attribute, L.3).

This attribute is defined for all elements that have the **%baseatts;** (B.2).

lang= L.3

Specifies the natural language (English, French, etc.) of the element's text content (**Lang** is also an element, L.2). Values should be taken from the standard language abbreviations provided in ISO 639 (Table 12-1).

```
lang CDATA #IMPLIED
```

Default value: undefined, meaning that the language may be anything.

Table 12-1 ISO-639 codes for selected languages

Code	Language Name (English/francais/native)
AR	Arabic/arabe/Arabi
DA	Danish/danois/Dansk
DE	German/allemand/Deutsch
EL	Greek/grec/Ellinika
EN	English/anglais

Table 12-1 ISO-639 codes for selected languages (Continued)

Code	Language Name (English/francais/native)
ES	Spanish/espagnol/Espanol
FI	Finnish/finnois/Suomi
FR	French/francais
GA	Irish/irlandais/Gaeilge
HI	Hindi
IN	Indonesian/indonesien
IS	Icelandic/islandais/Islenzk
IT	Italian/italien/Italiano
IW	Hebrew/hebreu/Iwrith
JA	Japanese/japonais/Nihongo
KO	Korean/coreen/Choson-o
NL	Dutch/neerlandais/Nederlands
NO	Norwegian/norvegien/Norsk
PL	Polish/polonais/Polski
PT	Portuguese/portugais/Portugues
RU	Russian/russe/Russkij
SV	Swedish/suedois/Svenska
TR	Turkish/turc/Turkce
UK	Ukrainian/ukrainien/Ukrainska
VI	Vietnamese/vietnamien/Tieng Viet-nam
ZH	Chinese/chinois/Zhongwen

langs= L.4

Specifies what languages (specified by the **`lang`** attribute, L.3) should be included in a **`Contents`** list or an **`Index`**. See **`Contents`** (L.5) for explanation.

```
langs CDATA #IMPLIED
```

The default is for every language to be included. The special value "UNSPEC" means any element whose language value is unspecified. Other values should be taken from the standard language abbreviations provided in ISO 639 (Table 12-1).

lastsep= L.5

How the last two items of a **`SimpleList`** with **`arrange="inline"`** should be separated. See the example at **`SimpleList`** (S.15).

```
lastsep CDATA #IMPLIED
```

There is no default, although ", and " could be one.

layer= L.6

The layer (from "0" for the top to any positive integer for the bottom) of the plane on which the **Frame**, **Graphic** or **Object** resides; that is, its Z dimension, or depth, relative to anything else that might overlap its X and Y coordinates.

```
layer NMTOKEN #IMPLIED
```

Comment

Situations creating a need for this attribute are to be strongly discouraged until browsers have more-sophisticated graphics-handling capabilities. Today, most cannot handle two graphics within the same frame even if they don't overlap.

"left", align= (A.4)

levels= L.7

Specifies what nested levels of an element should be listed in a **Contents** list or an **Index**. Typically, a **Contents** list would specify something like **elements="Section" levels="1 2 3 4"** to say that the first four levels of Sections should be listed. See **Contents** (C.22) for more explanation. A rendering application should always know the nesting level of each element (it probably doesn't matter for any element except **Section**).

```
levels NMTOKENS #IMPLIED
```

The default is for every level to be included. The values "1", "1 2", "1 2 3", "1 2 3 4", "1 2 3 4 5", and "1 2 3 4 5 6" are the only ones likely to be recognized.

<LI> L.8

An ordinary list item within an **OrderedList** (O.3), an **UnorderedList** (U.1), a **SegLI** segmented-list item (S.5), or a **VarLI** or **DistVarLI** variable-list item (V.3). **SimpleList** (S.15) has a different kind of item. The list mark (numeral, bullet, etc.) for **LI**s in an ordered or unordered list is specified in the parent list or list-group element (for instance, **OrderedList** or **OrdListGrp** (O.5)).

```
<!ELEMENT LI - - (Title?,(P|S|Annote|Frame|Danger|Caution|Warning|Admon
          |Table|OrderedList|UnorderedList|VariableList|SegmentedList
          |Listing|Pt|IndexTerm|Graphic|Object|ExternalText)*) >
<!ATTLIST LI
          %baseatts;
          %identity;
          %commonatts; >
```

Values of **type** that should be recognized: "Step", "Action", "Result", "Procedure", "Test", "Decision" (all for procedural docs), and "Definition"; **type** might be useful only for **SegmentedList**. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), and **%commonatts;** (C.19) for more on the attributes.

<Link> L.9

Element providing a general link structure. It contains exactly two linkends — ID references to elements within the working document. The first should be the link end from which traversal of the link would normally start: for instance, the text that references an annotation. With HyTime **NameLoc** elements (N.3) as intermediates, either or both link ends can be outside the current document.

```
<!ELEMENT Link - O  EMPTY>
<!ATTLIST Link
          %baseatts;
          %identity;
          remap.anchrole NMTOKENS #IMPLIED
          remap.extra NMTOKENS #IMPLIED
          remap.intra NMTOKENS #IMPLIED
          remap.refrange CDATA #IMPLIED
          linkends IDREFS #REQUIRED
          HyTime NMTOKEN #FIXED "ilink"
          anchrole NMTOKENS #FIXED "from to"
          extra NMTOKENS "A A"
          intra NMTOKENS "A A"
          refrange CDATA #FIXED "#ALL X">
```

Values of **type** have not yet been established. See the entries for the entities **%baseatts;** (B.2) and **%identity;** (I.3) for more on this element's attributes.

linkends= L.10

Attribute for **Link** (L.9) providing exactly two ID references for the elements at the ends of the link.

```
          linkends IDREFS #REQUIRED
```

The contents of the attribute must be two valid ID values, enclosed together in one pair of quotes and separated by white space.

links (HyTime module) — HyTime (H.4)

<ListHead> L.11

Wrapper for headings over the columns of a list (**OrderedList**, **SegmentedList**, **UnorderedList**, or **VariableList**).

```
<!ELEMENT ListHead - - (Title+) >
<!ATTLIST ListHead
          %baseatts; >
```

See the entry for the entity **%baseatts;** (B.2) for all this element's attributes. Use of the **type** attribute is not supported.

Nothing except order determines the position of the enclosed **Titles**. The first will be over the item numbers or marks, if defined (and not **numeration="nonum"** or **mark="nomark"**), and the second will be over the

next distinct column: the item text for **OrderedList** or **UnorderedList**, the **ListTerm**s of a **VariableList**, or the first **LI**s in each **SegLI** of a **SegmentedList**. There can be a third **Title** for a **VariableList**, and as many more as there are more **LI**s in a **SegmentedList**.

Style Note

The only way to skip a column, should that ever be desirable, would be to make a **Title** empty. If there should happen to be more **Title**s than columns, wrap around and start another row. A more likely problem is a **Title** too wide for the column; if possible, it should wrap within the column space as if in a table cell, but otherwise it will simply have to encroach on the next column.

<Listing> L.12

Not a list, but a display of program code or other "literal" text. It is not an in-line element: for a filename or a short example of code, input, or output in running text, use **T** with an appropriate type value. A listing may be multiple lines, or each line may be a separate **Listing** element (so that individual lines can be numbered). A Frame enclosing one or more **Listing**s may also be numbered (for example, **<Frame type="Exhibit" numeration=arabic label="Exhibit 2-1.">**).

The attribute **width** can be used to let recipients know how many characters should fit on one line (**width="80"** for typical character-based terminal screens). The entity **&newline;** can be used to indicate line ends.

Listings should be assumed to be "preformatted": all white space and line breaks should be preserved, but entity references and subelements should be recognized. A constant-width font should normally be used.

```
<!ELEMENT Listing  - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef
          |IndexTerm|Pt|Graphic|Object|ExternalText)* >
<!ATTLIST Listing
          %baseatts;
          %identity;
          %numbering;
          %commonatts;
          %textatts;
          width NUMBER #IMPLIED >
```

Values of the **type** attribute that should be recognized: "Example"; "Code"; "Input"; "Output"; "ScreenImage" (a whole screen); "ScreenText" (one line from a screen). See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), **%commonatts;** (C.19), and **&textatts;** (T.12) for more on this element's attributes.

"Listing", type=Frame (F.8)

liststyle= L.13

Information on the formatting of a **SegmentedList**, an analog of the CALS Table DTD's **tabstyle** attribute. No specific values have been established.

```
liststyle NMTOKEN #IMPLIED
```

<ListTerm> L.14

The headword of a **VariableList** item (V.2), such as a glossary term that is followed by a definition (which is in an **LI**).

```
<!ELEMENT ListTerm - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef|Pt
          |IndexTerm|Graphic|Object|ExternalText|Equation
          |GraphicalText)* >
<!ATTLIST ListTerm
          %baseatts;
          %identity;
          %commonatts;
          %textatts; >
```

Values of **type** that should be recognized: "GlossTerm"; "Acronym"; "Value"; "Fault"; "SysParam"; "ErrorCode"; "MessageCode"; "MenuItem". See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%commonatts;** (C.19), and **&textatts;** (T.12) for more on this element's attributes.

locs (HyTime module) — HyTime (H.4)

locsrc= L.15

Applies to **TreeLoc** (T.22)and **DataLoc** (D.2) only. The **locsrc** is an IDREF that refers to either a **NameLoc** (N.3) for external links, or an element for internal links. See **DataLoc** for examples.

```
locsrc IDREFS #IMPLIED >
```

"loweralpha", numeration= (N.15)

"lowerroman", numeration= (N.15)

M (mark — MTextType)

mark= M.1

The mark for items in an unordered list. There's no default, so a rendering application must be prepared to choose bullets for highest-level lists, dashes for nested lists, etc., or follow some other convention when the mark is not specified. Authors should realize that rendering applications are also free to override the specified mark to provide consistency in presentation. See also **numeration** (N.15) and **label** (L.1).

The mark is applied to **LI** list items (L.8), but is specified in the parent element, either an **UnorderedList** (U.1) or **UnordListGrp** (U.2). When authors want to emphasize a particular item in a list by changing its mark, they can define subsections of the list with additional **UnordListGrp** elements, giving a different mark to the item(s) in each group.

```
mark CDATA #IMPLIED
```

Since some rendering applications can't distinguish a null attribute value, "", from an undefined one, the special value "nomark" should be interpreted as an explicit "", meaning use no mark rather than choosing a different default.

Style Note

Although the mark is defined as CDATA notation, entity references like **•** (bullet) and **—** (em dash) should be replaced by the referenced characters during processing (as if the notation were RCDATA); this is inconsistent but correct behavior according to the SGML standard. However, it is a heavy burden on rendering applications to find and replace character entity references anywhere in an attribute value. Therefore it is strongly recommended that **mark** contain either a single entity reference and no other characters or else a string that does not contain an entity reference.

marked section M.2

Marked sections may occur in any SGML document instance (they are not mentioned in the TIM DTD but may be used in TIM documents). The content of the marked section is treated one of four ways:

- It is ignored, if the keyword IGNORE is used
- It is treated as regular content (as if there were no marked-section declaration) if the keyword INCLUDE is used. (This may seem useless, but the keyword can be provided by an entity reference whose value, "INCLUDE" or "IGNORE", can be changed like an on-off switch to suit different circumstances.)
- It is treated as CDATA (C.4) if the keyword is CDATA: the markup characters "<" and "&" are ignored, so no markup at all is recognized until the marked-

section-end delimiter occurs ("]]>"; it is important that that particular 3-character sequence not occur in the marked section itself).

- It is treated as RCDATA (R.1) if the keyword is RCDATA: the markup character "<" is ignored (no elements, processing instructions, or comments are recognized) but "&" is still recognized as the start of an entity reference, and entities will be substituted for. ("]]>"still must not occur in the marked section). XML does not allow RCDATA.

A marked section must be syntactically valid, meaning that the document must parse correctly if the processor observes the keyword: ignoring the section if the keyword is IGNORE, including it if it is INCLUDE, or ignoring the markup if it is CDATA or RCDATA. The section may contain anything from a single character to an entire document as long as it follows that rule.

Example 1: If you want to illustrate SGML markup in text, you must protect it from being interpreted as markup in one of three ways:

A. Put the text in a separate file and call it into the document with an entity reference.

```
<!ENTITY SGMLsample SYSTEM "../text/sample01.txt NDATA TEXT

...

SGML uses markup like this: &SGMLsample;.
```

B. Substitute character entity references for the markup characters "<" and "&".

```
SGML uses markup like this: <CopyrightNotice>&copy; 1997
ATIS. All rights reserved.</CopyrightNotice>
```

C. Enclose the example in a CDATA marked section.

```
SGML uses markup like this: <![ CDATA [
<CopyrightNotice>&copy; 1997 ATIS. All rights reserved.</
CopyrightNotice>]]>
```

Example 2: If you want to include parallel versions of the same content in one file, but have the reader see only the appropriate one, you can use multiple marked sections with an INCLUDE/IGNORE switch, changing the entity declarations when the circumstances change:

```
<!ENTITY Model400-110V "INCLUDE">

<!ENTITY Model400-220V "IGNORE">

...

... voltage should read between <![ &Model400-110V; [100 and
120]]><![ &Model400-220V; [215 and 240]]>V.
```

<MarkList> (explicit in TIM 1 only) M.3

A HyTime element in TIM 1 that contained marker pairs that delineated start and end points of a link end specified by a **DataLoc**. Its explicit use is never required for HyTime compatibility, so it has been dropped from TIM 2, but its function remains: the numbers in **TreeLoc** and **DataLoc** are implicitly MarkLists.

MarkLists contain a list of markers (numbers) that point to the position of an element to be linked in a **TreeLoc**, or sets of markers that specify left and right endpoints of a data stream in a **DataLoc**.

In a **TreeLoc** (T.22), each marker in the list refers to a node on a successive level of the document tree structure. For example, suppose that a section (referenced by the **NameLoc** referenced by this **TreeLoc**) contained a **TitleGroup** and several subsections, and the second subsection contained a **TitleGroup** and several paragraphs. The following MarkLists would be used to point to the fourth paragraph in the second subsection:

```
<TreeLoc>1 3 5</TreeLoc>
```

The first number is the count of siblings from the root section: 1 means that section itself. The second subsection is the third child of that section (the **TitleGroup** must be counted), and the subsection's fourth paragraph is its fifth child.

In a **DataLoc** (D.2), the first marker specifies the position of the left **quantum** (by default, a word; see Q.1) in a data stream and the second marker specifies the position of the right quantum in the data stream. The meaning of each marker depends upon whether its value is positive or negative. For example, take the following sentence:

```
The quick brown fox jumps over the lazy dog.
```

The string "fox jumps over" could be represented by all of the following marker combinations:

4 3 string begins at the fourth position from the left end of the axis and continues for 3 words.

4 -4 string begins at the third position from the left end of axis and continues until the fourth position from the right end of axis.

-6 3 string begins at the sixth position from the right end of the axis and continues for three words.

-3 -4 string begins at the third position leftward from the second marker's position, which is in the fourth position from the right end of the axis.

<MarkTitle> (TIM 1 only)

Now the first **Title** in a **ListHead** (L.11).

"MathConstant", "MathFunction", "MathVariable", type= — T (T.1)

%memos; M.4

An entity reference defined in TIM to allow elements appropriate for technical memos (such as address and signature blocks) to be added. They will be if the document prologue contains an entity declaration for **%memos;** that points to an actual file (the default definition is a null string). Example:

```
<!ENTITY % memos SYSTEM "../../lib/memos.dtd" NDATA SGML>
```

It is assumed that some of the declarations in the file would be for new elements, and the rest would redefine elements already in the TIM DTD so that they would be allowed to contain the new elements. Since TIM reads these declarations (if they exist) before its own, the new ones will be in effect. There are seven entity references for extensions to TIM 2, and their placement (before all element declarations in the DTD) and order (**%other;**, **%memos;**, **%rqmts;**, **%procs;**, **%eqns;**, **%flows;**, **%tables;**) are significant, since the order is the priority for redefining elements: only the first element and attribute declarations for a given element are used. The order shown reflects the likelihood that declarations in one extension would be redefining the elements in others: the table extension (already provided) is, of course, the least likely to affect others.

By default, **%memos;** is a null string, so its occurrence in the DTD does nothing:

```
<!ENTITY % memos        "">
%memos;
```

"MenuItem", type= — ListTerm (L.14), T **(T.1)**

"Message", type= — DistVarLI (D.7)

"MessageCode", type= — ListTerm (L.14)

"middle", valign= (V.1)

MIDI (notation) M.5

Notation that should be declared for a `filename.mid` (Musical Instrument Digital Interface) file.

```
<!NOTATION MIDI      SYSTEM "MIDI">
```

Example:

```
<!ENTITY clip001 SYSTEM "../midi/clip001.mid" NDATA MIDI>

...

<Object entityref="clip001">
```

The file should be stored in a `midi` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s). MIDI

files may be included in TEDD packages experimentally. Preferred formats will be specified in future TCIF Guidelines. Currently the options for sound files are AU (A.16), MIDI, and WAV (W.2).

MODULE

A module of the HyTime standard. Three processing instructions in TIM identify the three modules implemented in TIM. See HyTIme (H.4)

morerows= M.6

Applies to **`Entry`** only. Specifies the number of rows below the current one that the cell spans. The default is zero.

```
morerows NMTOKEN "0"
```

MOV (notation) M.7

Notation that should be declared for a `filename.mov` QuickTime movie file.

```
<!NOTATION MOV       SYSTEM "MOV">
```

Example:

```
<!ENTITY clip001 SYSTEM "../mov/clip001.mov" NDATA MOV>
```

The file should be stored in a `mov` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM document. MOV files may be included in TEDD packages experimentally. Preferred formats will be specified in future TCIF Guidelines. Currently the options for video files are Animator (no sound, A.8), AVI (A.17), MOV, and MPEG (M.8).

The QuickTime standard is a vague one. A `.mov` file should be tested to see that can be played on an MS-Windows PC known to be able to play at least some such files. If this cannot be demonstrated, change the filename extension to `.qt`, the directory to `qt`, and the notation to QUICKTIME (Q.2).

MPEG (notation) M.8

Notation that should be declared for a `filename.mpg` (Motion Picture Experts Group format) video file.

```
<!NOTATION MPEG      SYSTEM "MPEG">
```

Example:

```
<!ENTITY clip001 SYSTEM "../mpeg/clip001.mpg" NDATA MPEG>
```

The file should be stored in an `mpeg` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM document. MPEG files may be included in TEDD packages experimentally. Preferred formats

will be specified in future TCIF Guidelines. Currently the options for video files are Animator (no sound, A.8), AVI (A.17), MOV (M.7), and MPEG.

<MTextType> M.9

The **MTextType** element, a required subelement of **DocID**, contains the exact version number of TIM used when creating the **TDoc** instance:

```
<MTextType>TIM-2.0.0</MTextType>
```

MTextType replaces the one required **Status** element in TIM 1, which had the same content.

```
<!ELEMENT MTextType    - -  (#PCDATA)* >
<!ATTLIST MTextType
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

N (NAME — NUTOKEN)

NAME (SGML reserved name) N.1

"Name" has several meanings in SGML, all related. When the word is used in a DTD, it is used in attribute declarations, and it means that the attribute value is allowed to be anything that follows SGML's rules for NAMEs: it must begin with either an uppercase or lowercase alphabetical character (A-Z or a-z) and be composed only of alphanumeric characters, dashes, and/or periods (A-Z, a-z, 0-9, -, .). A NAME cannot contain any white space. The following examples are all valid NAMEs:

```
Table3-2

Section3.1.1

BR-123-456-789
```

In TIM 1, many attributes had name values, but in TIM 2 they are declared to be NMTOKENs (name tokens) for compatibility with XML. NMTOKENs are like NAMEs except that they may start with any of the characters valid in NAMEs, not just alphabetical characters. These are not valid NAMEs but are valid NMTOKENs:

```
-Figure-1-

.com

123-456-789-BR
```

The following examples are not valid NAMEs or NMTOKENs:

```
Table 3-2

Section3/1/1

123-456-789(BR)
```

There is also a length limit for NAMEs: the SGML default is 8 characters, but the TCIF SGML declaration increases the limit to 64. IDs (I.2) and IDREFs (I.4) are special kinds of NAMEs (implication: they cannot be purely numeric, they must start with a letter). The names of elements, attributes, and entities must also follow the rules for NAMEs.

"Name", type= — (I.9), IndexTerm (I.11) and T (T.1)

"name", quantum= (Q.1)

nameend= N.2

Applies to **SpanSpec**, **Entry**, and **EntryTbl**. Names the last column a cell should span (**Namest**, (N.5), names the first). Values for **Entry** (or **EntryTbl**) are inherited from **SpanSpec**. For **SpanSpec**, the definition is:

```
nameend NMTOKEN #REQUIRED
```

For **Entry** and **EntryTbl**, the definition is

```
nameend NMTOKEN #IMPLIED
```

<NameLoc> N.3

HyTime element used to complete links to external targets. When creating an external link, a **Ref** (R.2) is used to point to the **NameLoc** element (sometimes via a **DataLoc**, D.2, or **TreeLoc**, T.22). The **NameLoc**, in turn, points to the external element or entity, using a **NmList**. See examples at **NmList** (N.8).

```
<!ELEMENT NameLoc - - (NmList+) >
<!ATTLIST NameLoc
          id ID #REQUIRED
          HyTime NMTOKEN #FIXED "NameLoc">
```

NAMES (SGML reserved name) N.4

In an attribute declaration, this means that the value may be one or more NAMEs (N.1), with each NAME separated by a white space.

namest= N.5

Applies to **SpanSpec**, **Entry**, and **EntryTbl**. Names the first column a cell should span (**Nameend**, (N.2), names the last). Values for **Entry...** are inherited from **SpanSpec**. For **SpanSpec**, the definition is:

```
namest NMTOKEN #REQUIRED
```

For **Entry** and **EntryTbl**, the definition is

```
namest NMTOKEN #IMPLIED
```

nametype= N.6

Applies to **NmList** (N.8). Specifies the type of object the being named in the **NameLoc**. "Unified" is part of the standard architectural form of this attribute in the HyTime standard, but its use is not explained. The other two values are used in the examples at **NmList**.

```
nametype  (entity|element|unified) "element"
```

NDATA (SGML reserved name) N.7

"Non-SGML data." This is a data content type declared for external entities such as graphic files. It means that the data in the file are meaningless to an SGML parser and should not be parsed.

<NmList> N.8

HyTime element that names the external entities/elements to be linked in a **NameLoc**.

```
<!ELEMENT NmList - - (#PCDATA)>
<!ATTLIST NmList
          HyTime NMTOKEN #FIXED "NmList"
          nametype (entity|element|unified) "element"
          obnames (obnames|nobnames) "nobnames"
          docorsub ENTITY #IMPLIED
          dtdorlpd NMTOKENS #IMPLIED >
```

The **nametype** attribute is used to indicate whether the named item is an element (in an SGML document) or entity (a whole document, or a non-SGML file). If the item is an element, the content of **NmList** is the element ID; if the item is an entity, the content of **NmList** is the entity name. The **docorsub** attribute is used to identify the external document in which the ID named in **NmList** occurs, or in which the entity named in **NmList** is declared.

When **nametype="entity"**, the entity named in the content of **NmList** need not be declared in the current document, but if it is not, the document in which it is declared must have its own entity name, and that name must be declared in the prologue of the current document.

When **nametype="element"**, there is no choice. The document containing the element must have an entity name, and that name must be declared in the prologue of the current document.

There are three possible forms of a **NameLoc**-**NmList** link. First, to an element:

```
<!DOCTYPE TDoc PUBLIC "-//USA/TCIF//DTD TIMM-1//EN" [

...

<!ENTITY Tedd SYSTEM "../../../ipi95001/mtext/main.tim" NDATA
SGML>

...

]>

<TDoc>

...

...<Ref rid="TCIF-IPI-95-004-XREF-2"> (see "The TEDD
Package")</Ref><NameLoc id="TCIF-IPI-95-004-XREF-2"><NmList
docorsub="Tedd">TCIF-IPI-95-001-XREF-342</NmList></NameLoc>
```

Here "TCIF-IPI-95-001-XREF-342" is the ID of an element in the TEDD document.

Second, to an external entity declared in another document:

```
<!DOCTYPE TDoc PUBLIC "-//USA/TCIF//DTD TIMM-1//EN" [

...
```

```
<!ENTITY Tedd SYSTEM "../../../ipi95001/mtext/main.tim" NDATA
SGML>

...

]>

<TDoc>

...

...<Ref rid="TCIF-IPI-95-004-XREF-2"> (see Figure 1 of "The
TEDD Package")</Ref><NameLoc id="TCIF-IPI-95-004-XREF-
2"><NmList nametype="entity" docorsub="Tedd">fig001</NmList></
NameLoc>
```

Here "fig001" is the entity name declared in the prologue of the "Tedd" document. It doesn't matter whether there's also a "fig001" declared in the current document.

And third, to an external entity declared in the current document:

```
<!DOCTYPE TDoc PUBLIC "-//USA/TCIF//DTD TIMM-1//EN" [

...

<!ENTITY TEDDfig1 SYSTEM "../../../ipi95001/tiff/fig001.tif"
NDATA TIFF>

...

]>

<TDoc>

...

...<Ref rid="TCIF-IPI-95-004-XREF-2"> (see Figure 1 of "The
TEDD Package")</Ref><NameLoc id="TCIF-IPI-95-004-XREF-
2"><NmList nametype="entity">TEDDfig01</NmList></NameLoc>
```

In this case, **docorsub** isn't used, so the entity declaration is found in the current document. The latter two uses are not quite equivalent to

```
...

...<Graphic entityref="TEDDfig01">
```

since they present the figure indirectly, by cross-reference. There is also the consideration that some browsers may understand direct entity references but not HyTime links to entities, or even vice-versa.

NMTOKEN (SGML reserved name) N.9

A NMTOKEN is almost the same thing as a NAME (N.1): it consists of alphanumeric characters, dashes, and/or periods (A-Z, a-z, 0-9, -, or .). Unlike a NAME, a NMTOKEN can begin with any of the valid characters. Attributes that have a value

type of NMTOKEN (such as **colname** and **spanname**) may, for convenience, be either words or numbers or hybrid: "first", "1", or "col1", for instance.

"nobnames", obnames= **(O.2)**

"nocatres", catres= **(C.1)**

"nocatsrc", catsrc= **(C.2)**

"noinfix", infix= **(I.12)**

"noinherit", inheritnum= **(I.14)**

"nolabel", label= **(L.1)**

"nomark", mark= **(M.1)**

"none", frame= **(F.9)**

"noprefix", prefix= **(P.15)**

"norev", revstat= **(R.16)**

"norm", quantum= **(Q.1)**

"normal", present= **(P.16)**

"nosplit", splitnum= **(S.21)**

"nosufix", suffix= **(S.28)**

"Note", type= — Admon **(A.2)**, Annote **(A.9)**

NMTOKENS (SGML reserved name) N.10

NMTOKENS means multiple NMTOKENs (N.9): an attribute defined with the value type NMTOKEN can have only one, but if the value type is NMTOKENS, there can be any number of NMTOKENs, separated by white space.

NOTATION (SGML reserved name) N.11

Reserved name used to indicate that an attribute's value is a notation. A notation is a flag that indicates parts of a document that are in a data format other than SGML. One TIM attribute, **format**, has a NOTATION value. Its use alerts the rendering application that some process must be invoked to render the non-SGML entity (TIFF, EQN, HTML, etc.), such as the launching of a graphics viewer.

(Number sign or pound sign

"#" is put in front of SGML reserved names whenever they occur where user-defined names are also allowed. In this document, look up the name without the "#": for instance, PCDATA (P.3).

NUMBER (SGML reserved name) N.12

Reserved name for an attribute value type in which all the characters are digits (0-9). Attributes that had (in TIM 1) a value type of NUMBER (such as **cols**, **segments**, and **restartsat**) are used to provide numerical data needed to properly display a given element. They are also used as a workaround in some DTDs, including the CALS Table DTD that is part of TIM, because an element cannot have two attributes with the same predefined NAME values, like "yes" or "no". Not allowed:

```
pgwide    (yes|no)    #IMPLIED
rowsep    (yes|no)    #IMPLIED
```

Allowed:

```
pgwide    NUMBER    #IMPLIED
rowsep    NUMBER    #IMPLIED
```

Where "1" and "0" stand in for "yes" and "no."

XML requires that NMTOKEN (N.9) be used in place of NUMBER in declaring attribute value types, so NMTOKEN is used in TIM 2.

%numbering; N.13

An entity defining numbering attributes available for elements that can reasonably be labeled with numbers, letters, or some other predictable sequence of characters:

```
<!ENTITY % numbering
          "numeration (nonum|arabic|outline|upperalpha|loweralpha
             |upperroman|lowerroman|symbol) #IMPLIED
           prefix CDATA #IMPLIED
           infix CDATA #IMPLIED
           suffix CDATA #IMPLIED
           inheritnum (inherit|inherit0|inherit1|inherit2|inherit3
             |noinherit) #IMPLIED
           inheritfrom NMTOKENS #IMPLIED
           splitnum (nosplit|splitarabic|splitualpha|splitlalpha
             |spliturоman|splitlroman|splitoutline) #IMPLIED
           continuation (continues|restarts|holds) #IMPLIED
           restartsat NMTOKEN #IMPLIED
           increment NMTOKEN #IMPLIED
           shownum NMTOKEN #IMPLIED
           symbolseq CDATA #IMPLIED">
```

Whenever "%numbering;" appears in an element's ATTLIST, the twelve attribute declarations above are substituted for it. Refer to the individual entries for the

meanings and uses of each: **numeration** (N.15), **prefix** (P.15), **infix** (I.12), **suffix** (S.28), **inheritnum** (I.14), **inheritfrom** (I.13), **splitnum** (S.21), **continuation** (C.23), **restartsat** (R.15), **increment** (I.8), **shownum** (S.13), and **symbolseq** (S.34).

NUMBERS (SGML reserved name) N.14

Reserved name for a list of NUMBERs, with each number separated by a white space. The only attribute in TIM 1 that had a value type of NUMBERS was **levels**, which was used to specify what nested levels of an element should be listed in a **Contents** list or **Index**. In TIM 2 this value type was changed to NMTOKENS (N.10) to conform to the XML standard.

numeration= N.15

Specifies the type of numbering/lettering for a sequence of items such as sections, tables, numbered paras, or items in an **OrderedList**. No default is defined, so if the numeration is not specified, a rendering application must be prepared to choose arabic numerals for highest-level lists, uppercase letters for lists nested within those, etc., or follow some other convention. Authors should realize that rendering applications are also free to override the specified numeration to provide consistency in presentation. It is likely that simple browsers will use the label provided for each element, while applications that recombine and reuse information will ignore the label and instead use the algorithm implied by **numeration** and related attributes.

Prefixes and suffixes may be specified with **prefix** (P.15) and **suffix** (S.28). Numbering normally restarts for each new list, and continues for each section, paragraph, distributed-list item, etc. Those defaults can be overridden with **continuation** (C.23) and **restartsat** (R.15). See also **mark** (M.1) and **label** (L.1). Concatenation of higher-level numbers (e.g., "Section 2.5.3") can be specified with **inheritnum** and **inheritfrom**, and the character between the concatenated numbers can be specified with **infix**, which works like **prefix** and **suffix**.

List-item numbering is defined at the list or list-group level (that is, in the **...List ...ListGrp** element), not the item level, but it can be changed within the list (e.g., 1 ... 2 ... 3a ... 3b ... 4) by using more than one list group; see **OrdListGrp** (O.5), **SegListGrp** (S.6), and **VarListGrp** (V.4). By the way, the 3a ... 3b effect is done with yet another attribute, **splitnum** (S.21).

```
"numeration (nonum|arabic|outline|upperalpha|loweralpha
   |upperroman|lowerroman|symbol) #IMPLIED
```

The **numeration** attribute is defined, for the elements that have it, through the entity reference **%numbering;** (N.13).

The default value for **numeration**, though undefined, should be assumed to be "nonum" for all elements except **LI**'s within an **OrderedList**, where it should

be assumed to be "arabic", at least for the first hierarchical level of lists. But authors should realize that rendering applications are free to set their own defaults for numbering styles for the sake of consistency in presentation, and may even override explicitly stated values. For this reason, cross-referencing by text only ("see Section 2.1") should be avoided in favor of hypertext links and/or system-generated textual references.

The choice of "outline" for numeration leaves open how each level of the outline is labeled. One possibility is I. A. 1. a. (1) (a) (i). The choice of "symbol," on the other hand, requires specification of the sequence of symbols to be used, with the **`symbolseq`** (S.34) attribute.

NUTOKEN (SGML reserved name) N.16

Reserved name for an attribute value type that always begins with a digit (0-9), and consists of alphanumeric characters, dashes, and/or periods (A-Z, a-z, 0-9, -, or .). Attributes that have a value type of NUTOKEN (such as **`charoff`** and **`width`**) are used to specify some form of measurement (e.g., 1cm, 2.5in, etc.). TIM 2, to conform to XML, uses NMTOKEN (N.9) in place of NUTOKEN.

O (Object — other)

<Object> O.1

An element referencing an external entity that is neither text nor ordinary graphic. It may be either the entire content of an enclosing element (normally a **Frame**) or an alternative representation (of a table, figure, etc.), in which case it should be referenced by another element's **altreps** attribute). An **Object** element references a non-ASCII file (audio, multimedia, encrypted text, etc.) other than a 2-dimensional, rectangular graphic image (GIF, CGM, EPS, etc.), which should be referenced by a **Graphic** element (G.2). The distinction is made because 2-D graphics generally can be rendered faithfully in line with text in a paged or scrolling presentation, while other objects cannot.

```
<!ELEMENT Object - O EMPTY >
<!ATTLIST Object
          %baseatts;
          %identity;
          %commonatts;
          dimensions NMTOKENS #IMPLIED
          placement NMTOKENS #IMPLIED
          valign (top|middle|bottom) #IMPLIED
          align (left|right|center) #IMPLIED
          layer NMTOKEN #IMPLIED
          application CDATA #IMPLIED
          entityref ENTITY #REQUIRED >
```

No values have been established for the **type** attribute. The **application** attribute may be used to specify what application is needed to view or play the file. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), and **%commonatts;** (C.19) for more on this element's attributes.

obnames= O.2

Applies to **NmList** (N.8). A value of "obnames" causes any content of the target element that is the same as an **ID** (I.2) to be interpreted as an IDREF by HyTime instead of text, adding an extra jump to the cross-reference. A value of "nobnames" causes all element content to be treated as text by HyTime — that is, as the actual endpoint of the link.

```
          obnames (obnames|nobnames) "nobnames"
```

<OrderedList> O.3

An ordinary list in which items are sequential (and probably sequentially numbered or lettered) — compare **UnorderedList** (U.1), **SimpleList** (S.15), **DistOrdLI** (D.6), **VariableList** (V.2), and **SegmentedList** (S.7). Its items may be grouped using the **OrdListGrp** element.

```
<!ELEMENT OrderedList - - ((Title|TitleGroup)?,ListHead?,(OrdListGrp+
```

```
          |(Danger|Caution|Warning|Admon|Annote|LI|Pt)+)) >
<!ATTLIST OrderedList
          %baseatts;
          %identity;
          %numbering;
          %commonatts; >
```

No values have been established for the **type** attribute. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), and **%commonatts;** (C.19)for more on this element's attributes.

<OrderInfo> O.4

Plain-text information in the **DocID** (D.27) about how to order the document.

```
<!ELEMENT OrderInfo - - (#PCDATA)* >
<!ATTLIST OrderInfo
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Specific forms for the information it contains are left to the document originator.

<OrdListGrp> O.5

A section of an **OrderedList** (O.3). It may have a title that distinguishes it from the other sections of the list or different headings for the mark and content of the list items within it, or it may specify a different list-number sequence for the items within it. **OrdListGrp** allows the list numbering, but not the list itself, to be discontinuous; a discontinuous ("distributed") list must use **DistOrdLI** items (D.6). **OrdListGrp** is analogous to **TGroup** within tables. The numbering of the items is specified by the numbering attributes in **OrdListGrp**. The group itself, though it can have a title, cannot have a mark or number.

```
<!ELEMENT OrdListGrp - - (Title?,ListHead?,(Danger|Caution|Warning|Admon
          |Annote|LI|Pt)*) >
<!ATTLIST OrdListGrp
          %baseatts;
          %identity;
          %numbering;
          %commonatts; >
```

Use of the **type** attribute is not supported. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), and **%commonatts;** (C.19) for all this element's attributes.

Style Note

It is a little awkward that there is no mechanism for making an item's label bold to highlight it, since writers sometimes do that. This seeming deficiency is a

consequence of the overall philosophy of structured documents. Item numbers are not content, just labels of convenience. The labels have no permanence or meaning — items can be renumbered in a different style at the whim of the user. If an item should be emphasized, emphasize the item, not its label. (On the other hand, there is no prohibition from using the **presspec** attribute to specify "boldlabel"; it just isn't likely to be interpreted by anyone else's application.)

orient= O.6

Applies to **Table** only. Specifies whether a table is intended to be in portrait or landscape orientation on an ordinary paper page. The default value is "Port." Most SGML-aware applications have no need to implement this attribute, since they are not dealing with a fixed page size.

```
orient (Port|Land) #IMPLIED
```

%other; O.7

An entity reference defined in TIM to allow an extension for some heretofore unimagined purpose. If such a purpose arises, the extension can be made by putting into the document prologue an entity declaration for **%other;** that points to an actual file (the default definition does not). Example:

```
<!ENTITY % other SYSTEM "../../lib/widgets.dtd" NDATA SGML>
```

It is assumed that some of the declarations in the file would be for new elements, and the rest would redefine elements already in the TIM DTD, so that they would be allowed to contain the new elements. Since TIM reads these declarations (if they exist) before its own, the new ones will be in effect. There are seven entity references for extensions to TIM 2, and their placement (before all element declarations in the DTD) and order (**%other;**, **%memos;**, **%rqmts;**, **%procs;**, **%eqns;**, **%flows;**, **%tables;**) are significant, since the order is the priority for redefining elements: only the first element and attribute declarations for a given element are used. The order shown reflects the likelihood that declarations in one extension would be redefining the elements in others: the table extension (already provided) is, of course, the least likely to affect others.

By default, **%other;** is a null string, so its occurrence in the DTD does nothing:

```
<!ENTITY % other        "">
%other;
```

"Output", type= — ListTerm (L.14) and T **(T.1)**

P (P — Publisher)

<P> P.1

Element used for paragraphs and paragraph-like structures. A paragraph may not contain another paragraph (a subparagraph) directly. Such structures are considered rare enough in telecom (and most other) documents that they can be dealt with as special cases. **Ps** within lists or structural groupings are valid.

A **P** can have a number but not a title. To title a para, enclose it in a **Section** or an **S** structure.

```
<!ELEMENT P - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef|SimpleList
          |IndexTerm|Pt|Graphic|Object|ExternalText|Flowchart|Equation
          |GraphicalText|S|Annote|Frame|Danger|Caution|Warning|Admon
          |Table|OrderedList|UnorderedList|VariableList|SegmentedList
          |Listing)*>
<!ATTLIST P
          %baseatts;
          %identity;
          %numbering;
          %commonatts;
          %textatts;>
```

Values of **type** that should be recognized: "Quote", "QuoteAttrib" (attribution for a quotation), and "Plain" (the default). Other values should be anticipated, as with the other very-general-purpose elements, **Pt**, **S**, and **T**. See the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), **%commonatts;** (C.19), and **&textatts;** (T.12) for more on this element's attributes.

Style Note

An indented paragraph that cannot be explained as a block quote or other distinct structural unit should probably be marked as an item of a **SimpleList** (S.15). A centered paragraph can be indicated with **presspec="centered"** (this may not be honored by the rendering application).

"Page", type= — T (T.1)

"Parameter", type= — T (T.1)

"Part", type= — Section (S.4)

"PartNum", type= — T (T.1)

PBM (notation) P.2

Notation that should be declared for a `filename.pbm` (Portable BitMap) graphic file.

```
<!NOTATION PBM       PUBLIC "+//ISBN 0-7923-9432-1::Graphic
          Notation//NOTATION Jef Poskanzer Portable Bit Map//EN">
```

Example:

```
<!ENTITY fig001 SYSTEM "../ppm/fig001.pbm" NDATA PBM>
```

The file should be stored in a `ppm` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM document. PBM files should not be included in TEDD packages that follow all TCIF Guidelines. Currently the TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG (P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18).

PCDATA (SGML reserved name) P.3

"Parsed character data," data characters that occur in marked-up text that are not recognized as markup by an SGML parser. For the most part, it means "plain text," but it is distinguished from RCDATA (R.1) and CDATA (C.4), in which some or all SGML markup is ignored. Except where CDATA or a special NOTATION (N.11) has been identified, all characters within a TIM document that aren't markup are PCDATA. Take, for example, the following string of marked-up text:

```
<P>All emphasized text should be <Emph type="ital">italicized</
Emph> and CAPITALIZED.</P>
```

The character strings "All emphasized text should be ", "italicized", and " and CAPITALIZED." are PCDATA and are treated as normal text by the SGML parser. The character strings "<P>", "<Emph type="ital">", "</Emph>", and "</P>" are not PCDATA because the parser recognizes them as SGML markup.

"#PCDATA" is found in the content models of all elements that can contain text directly. "#" is put in front of SGML reserved names whenever they occur where user-defined names are also allowed.

PCX (notation, TIM 1 only) P.4

Notation that should be declared for a `filename.pcx` (PC Paintbrush) bitmap graphic file.

```
<!ENTITY fig001 SYSTEM "../pcx/fig001.pcx" NDATA PCX>
```

The file should be stored in a `pcx` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM document. PCX files should not be included in TEDD packages that follow all TCIF Guidelines. Currently the TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG (P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18).

PCX files can be 1-bit (black-and-white), 4-bit (16-color), 8-bit (256-color), or 24-bit (16.7 million colors). Compression is variable, and usually not as good as in GIF, JPEG, PNG, and some kinds of TIFF files.

PDF (notation) P.5

Notation that should be declared for a `filename.pdf` Adobe Portable Document Format file.

```
<!NOTATION PDF       SYSTEM "PDF">
```

Example:

```
<!ENTITY fig001 SYSTEM "../pdf/fig001.pdf" NDATA PDF>
```

The file should be stored in a `pdf` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s). PDF files containing graphics may be included in TEDD packages experimentally. Preferred formats will be specified in future TCIF Guidelines. Currently the options for vector-graphic files and metafiles are CGM-BINARY (C.6), CGM-CLEAR (C.7), DXF (D.35), EPS (E.12), PDF (P.5), PICT (P.9), WMF (W.5), and WPG (W.6). Currently the TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG (P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18).

PGM (notation) P.6

Notation that should be declared for a `filename.pgm` (Portable GrayMap) graphic file.

```
<!NOTATION PGM       SYSTEM "PGM">
```

Example:

```
<!ENTITY fig001 SYSTEM "../ppm/fig001.pgm" NDATA PGM>
```

The file should be stored in a `ppm` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM document. PGM files should not be included in TEDD packages that follow all TCIF Guidelines. Currently the TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG (P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18).

pgwide= P.7

Applies to **`Table`** only. Specifies whether a table is intended to be expanded to the full text width of an ordinary paper page, rather than stay within one column of a two-column page. Rendering applications should assume a default value of "0" (no).

```
pgwide NMTOKEN #IMPLIED
```

"Phone", type= — T (T.1)

PIC (notation) P.8

A text notation containing **`troff pic`** drawing commands. (Other formats known as PIC are not accommodated in TEDD packages.)

```
<!NOTATION PIC       SYSTEM "PIC">
```

It may be within a TIM file:

```
<Listing format="PIC"> ....
```

Or it may be in a separate file with a `.pic` extension, in a `troff` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s):

```
<!ENTITY pic001 SYSTEM "../troff/pic001.pic" NDATA PIC>

...

<ExternalText entityref="pic001">
```

The PIC format may be included in TEDD packages experimentally. Preferred graphic formats will be specified in future TCIF Guidelines.

PICT (notation) P.9

Notation that should be declared for a `filename.pct` (Macintosh QuickDraw Picture format) graphic metafile (metafiles may contain drawings, bitmaps, and text).

```
<!NOTATION PICT      PUBLIC "+//ISBN 0-7923-9432-1::Graphic
          Notation//NOTATION Apple Computer QuickDraw Picture//EN">
```

Example:

```
<!ENTITY fig001 SYSTEM "../pict/fig001.pct" NDATA PICT>
```

The file should be stored in a `pict` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s). PICT files may be included in TEDD packages experimentally. Preferred formats will be specified in future TCIF Guidelines. Currently the options for vector-graphic files and metafiles are CGM (C.6), CGM-CLEAR (C.7), DXF (D.35), EPS (E.12), PDF (P.5), PICT (P.9), WMF (W.5), and WPG (W.6).

placement= P.10

Preferred horizontal and vertical offset (from the top left corner of the parent window to the center) of a **`Frame`**, **`Graphic`**, or presentation window for an **`Object`**.

```
placement CDATA #IMPLIED
```

Example for a graphic centered in a 640x480-pixel window:

```
<Graphic dimensions="200 100" placement="320 240" ...
```

Units of measure that should be understood: pixels (default, no abbreviation); "mm" (millimeter); "cm" (centimeter); "in" (inch); "p" (pica, 1/6 in), "pt" (point, 1/72 in); "dd" (didot, 1/67.431 in); "cc" (cicero, 1/5.619 in, 12 didots). Use lower case. Two units of measure cannot be combined in one value (e.g., "3p6" should be "3.5p" or "42pt").

"Plain", type= — P (P.1), T (T.1)

PNG (notation) P.11

Notation that should be declared for a `filename.png` (Portable Network Graphic) file.

```
<!NOTATION PNG       SYSTEM "PNG">
```

Example:

```
<!ENTITY fig001 SYSTEM "../png/fig001.png" NDATA PNG>
```

The file should be stored in a `png` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM document. Currently the TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG (P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18).

The PNG format is a recent invention intended to replace GIF, which uses a patented compression method that software vendors (not users) must pay royalties for. So far PNG is supported by fewer than half as many applications as GIF. PNG files can be 1-bit (black-and-white), 4-bit (16-color), 8-bit (256-color), 16-bit, 24-bit (RGB) or 32-bit (CMYK). Compression is comparable to GIF's LZW compression, typically about 65% with photographic images, so files can still be very large, but diagrams that are mostly white background can be compressed more than 95%.

"Policy", type= — Admon (A.2)

 # (Pound sign or number sign

"#" is put in front of SGML reserved names whenever they occur where user-defined names are also allowed. In this document, look up the name without the "#": for instance, PCDATA (P.3).

PPM (notation) P.12

Notation that should be declared for a `filename.ppm` (Portable PixMap) color graphic file.

```
<!NOTATION PPM       SYSTEM "PPM">
```

Example:

```
<!ENTITY fig001 SYSTEM "../ppm/fig001.ppm" NDATA PPM>
```

The file should be stored in a `ppm` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM document. PPM files should not be included in TEDD packages that follow all TCIF Guidelines. Currently the TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG (P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18).

PPT (notation) P.13

Notation that should be declared for a `filename.ppt` Microsoft PowerPoint presentation file.

```
<!NOTATION PPT       SYSTEM "PPT presentation">
```

Example:

```
<!ENTITY pres001 SYSTEM "../other/pres001.ppt" NDATA PPT>

...

<Object entityref="pres001">
```

The file should be stored in an `other` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s). PPT files may be included in TEDD packages experimentally. Preferred formats will be specified in future TCIF Guidelines.

"Preface", type= — Section (S.4)

prefer= P.14

Indicates that a particular **`IndexEntry`** element (I.10) marks the best reference for the indexed term. In print indexes this is normally rendered by setting the particular page number in bold type. In a hyperlinked index, its link should be the first one accessed for that index entry.

```
prefer (preferred|ordinary) #IMPLIED
```

prefix= P.15

An optional prefix for the number of a section, list item, frame, paragraph, etc. May include text: e.g., "Figure ". Authors should understand that a prefix and other presentation specifications may be overridden by the rendering application for the sake of consistency in presentation — for example, the author may specify "Figure " but the rendering application may produce "Exhibit ".

Note that separating space should be included at the end of a **prefix**, the beginning of a **suffix**, and both ends of an **infix**, since they will not be assumed by the rendering application. Separators should not be included at the beginning of **prefix**, the end of **suffix**, or either end of **label**. These are to be supplied by the rendering application.

```
prefix CDATA #IMPLIED
```

Default value: undefined, meaning the rendering application may choose its own prefix (the best choice will usually be nothing, but it may be organizational style to put "Chapter " or "Section " in front of first-level section numbers, and so on). Since some rendering applications can't distinguish a null attribute value, "", from an undefined one, the special value "noprefix" should be interpreted as an explicit "", meaning use no prefix rather than choosing a different default.

The **prefix** attribute is defined, for the elements that have it, through the entity reference **%numbering;** (N.13).

present= P.16

Presentation. Contains information about where and when to present the element.

```
present (normal|float|alert|altrep) #IMPLIED
```

This attribute is defined for all elements that have the **%baseatts;** (B.2).

Default value: "normal", meaning present the content where you find it in the data stream. The value "float" means (for an on-line presentation that can do this) hide the content until a link to it is activated (typically by user's clicking on a hotspot at the start of the link), then present it in a separate window. This is the on-line equivalent of a footnote or endnote.

The value "alert" means present the content in a separate window, but do not wait for the user to request it. As soon as the start of the link is visible, pop up the window. If an application cannot do this, it should present the content in where it occurs in the data stream, but do something to indicate its importance. Links to alerts use the **alerts** attribute.

The value "altrep" indicates an alternative or supplemental representation of an element's contents, which should not be displayed unless a link is activated. Links to altreps use the **altreps** attribute. The alternative representation could be another TIM representation using a different natural language (e.g., French rather than English) or ISO 646 (ASCII) text with another kind of notation (that is,

a different markup, like **troff eqn** or **html**), in which case the enclosed element's **`format`** attribute must specify the notation. An alternative representation outside the TIM file would be **`ExternalText`** (E.16), **`Graphic`** (G.2), or **`Object`** (O.1).

Comment

There are at least two other ways to present alternative versions of the same content, and any of the three may be appropriate. Marked sections (M.2) provide two or more alternative versions inside a TIM file, with a switch (an entity declaration) determining which one will be presented. Although the switch can be reset before processing, after loading into an application only one representation will be available to the user. For external entities, another technique is to provide only file basenames (without extensions like `.gif`, `.eps`, or `.tif`) in the entity references. Applications aware of this technique will choose a representation according to a table of preferences (something like, "use `.gif` if available, second choice is `.tif`, third choice is `.eps`"), saving the user from having to choose. Using **`altrep`**s does leave the choice to the user, which is sometimes desirable.

presspec= P.17

Presentation Specification. Contains information about how to present the element (its format).

```
presspec CDATA #IMPLIED
```

Default value: undefined. The only values agreed upon so far are "centered" and "right", for alignment of **`P`** paragraphs.

This attribute is defined for all elements that have the **`%baseatts;`** (B.2).

<Primary> (TIM 1 only)

Was a subelement of **`IndexEntry`** (I.10), no longer used.

"Procedure", type= — DistOrdLI (D.6), LI **(L.8)**, and SegmentedList **(S.7)**

"Procedures", type= — Contents (C.22), Index **(I.9)**

%procs; P.18

An entity reference defined in TIM to allow an extension for procedure markup. The extension can be made by putting into the document prologue an entity declaration for **`%procs;`** that points to an actual file (the default definition does not). Example:

```
<!ENTITY % procs SYSTEM "../../lib/procs.dtd" NDATA SGML>
```

It is assumed that some of the declarations in the file would be for new elements, and the rest would redefine elements already in the TIM DTD, so that they would be allowed to contain the new elements. Since TIM reads these declarations (if they exist) before its own, the new ones will be in effect. There are seven entity references for extensions to TIM 2, and their placement (before all element declarations in the DTD) and order (**%other;**, **%memos;**, **%rqmts;**, **%procs;**, **%eqns;**, **%flows;**, **%tables;**) are significant, since the order is the priority for redefining elements: only the first element and attribute declarations for a given element are used. The order shown reflects the likelihood that declarations in one extension would be redefining the elements in others: the table extension (already provided) is, of course, the least likely to affect others.

By default, **%procs;** is a null string, so its occurrence in the DTD does nothing:

```
<!ENTITY % procs        "">
%procs;
```

<ProdDsc> P.19

May occur any number of times within **DocID** (D.27) to provide information about the product(s) to which the document is relevant.

```
<!ELEMENT ProdDsc    - -  (#PCDATA)* >
<!ATTLIST ProdDsc
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Specific forms for the information it contains are left to the discretion of the document originator. Example:

```
<ProdDsc>A Component of Bellcore Intelligent Information
Services</ProdDsc>

<ProdDsc>Product Family: Enhanced Information Objects In Orbit
(EIEIO)</ProdDsc>
```

"ProductName", type= — T (T.1)

"Prompt", type= — T (T.1)

<PropMsg> P.20

```
<!ELEMENT PropMsg    - -  (#PCDATA)* >
<!ATTLIST PropMsg
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

<PropStat> P.21

May occur once within **DocID** (D.27) to provide information about the proprietary status of the document.

```
<!ELEMENT PropStat    - -  (#PCDATA)* >
<!ATTLIST PropStat
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Specific forms for the information it contains are left to the discretion of the document originator. Example:

```
<PropStat>Confidential - Addressee Only
BELLCORE PROPRIETARY - INTERNAL USE ONLY</PropStat>
```

PS (notation) P.22

```
<!NOTATION PS        SYSTEM "PS">
```

<Pt> P.23

Element used to mark physical points in a document. May be used as the target of a link, in which case a value must be given for the **id** attribute.

```
<!ELEMENT Pt - O EMPTY >
<!ATTLIST Pt
          %baseatts;
          %identity;
          %numbering;
          %commonatts;
          close NMTOKEN #IMPLIED
          spanend IDREF #IMPLIED
          HyTime NMTOKEN #FIXED "clink"
          HyNames CDATA #FIXED "linkend spanend"
          refrange NMTOKENS "spanend X" >
```

Expect that the **type** and **remap** attributes will be used frequently with **Pt** and given many different values. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), and **%commonatts;** (C.19) for more on this element's attributes.

Comment

Pt is not the best way to mark the endpoint of a link. It is better to put the ID on the highest-level element the link should point to, because the rendering application may do something like highlighting the portion of the document pointed to by the link. A **Pt** does not fully identify that relevant portion.

However, **Pt** may be the only option when creating links automatically, since the source document may contain only a point-like marker.

Comment

Pt provides a way to mark up an alternative hierarchy (tree structure). For instance, ordinary elements cannot be used to mark both the paragraphs and the verses in an English-language bible, because the structures overlap. This is not legal SGML:

```
...<P>...</Verse><Verse label="Acts 8:1">And Saul was there,
giving approval to his death.</P><P>On that day ...</
Verse><Verse label="Acts 8:2">...</P>...
```

This is legal:

```
...<P>...<Pt remap="Verse" label="Acts 8:1">And Saul was
there, giving approval to his death.</P><P>On that day ...<Pt
remap="Verse" label="Acts 8:2">...</P>...
```

And so is this:

```
...<Pt remap="P">...</Verse><Verse label="Acts 8:1">And Saul
was there, giving approval to his death.<Pt remap="P">On that
day ...</Verse><Verse label="Acts 8:2">...<Pt remap="P">
```

Similar situations can arise in marking up legacy documents that used idiosyncratic structural elements (e.g., requirements documents that labeled arbitrary portions of text as requirements).

When the structures in the alternative hierarchy are discontinuous (that is, where end-tag omission would cause ambiguity, as when requirement objects are interspersed with explanatory text), both beginning and end points of elements will have to be marked. TIM provides two mechanisms to do that, since it is not clear which one future applications might be able to implement. One is to use the **spanend** IDREF attribute in the starting **Pt** to identify the ending **Pt**: **<Pt remap="Foo" spanend="pt001">text<Pt id="pt001">**. The other is to use the **close** attribute to identify the element being closed at ending **Pt**: **<Pt remap="Foo">text<Pt close="Foo">**. Either of these should be understood to be equivalent to **<Foo>text</Foo>** in the alternative structure.

Comment

There is no rule prohibiting the use of **Pt** elements to mark and label the page breaks in the source document (**<Pt type="Page" label="Page 1-90">**), but this is a dubious practice, since discrete pages can't always, and maybe shouldn't, be identified in electronic documents. In well-structured hypertext, the pseudo-structures that page breaks create are superfluous at best.

PUBLIC (SGML reserved name) P.24

This contrasts with SYSTEM (S.35), and says that an identifier is to be looked up in a catalog of public identifiers rather than being left to be interpreted by the system. Single-use entities, like graphic files, are generally given SYSTEM identifiers, which either are, or can be converted to, pathnames that locate a file. Reused entities, like DTDs, are usually given PUBLIC identifiers. This is the way most TIM files begin:

```
<!DOCTYPE TDoc PUBLIC "-//USA/TCIF//DTD TIMM-1//EN" [
<!ENTITY fig001 SYSTEM "../gif/fig001.gif" NDATA GIF>
....
```

"PubDate", type= — T (T.1)

<Publisher> P.25

An optional subelement of **DocID** (D.27) identifying the document's publisher, if different from the originator identified in **Company** (C.20).

```
<!ELEMENT Publisher    - -  (#PCDATA)* >
<!ATTLIST Publisher
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Content and format are left to the document provider.

"Publisher", type= — T (T.1)

"PublisherNote", type= — Annote (A.9)

Q (quantum — QUICKTIME)

quantum= Q.1

The type of data token that is counted in a **DataLoc** (D.2). The possible values of **quantum** are: "str" (each character); "norm" (all but white space); "word" (SGML name characters, which include hyphen and period but not apostrophe or comma); "name" (SGML name tokens, which ignores strings that don't obey all the rules of SGML names, so "ABC" and "B-2" would be counted, but "123" and "2-B" would not); "sint" (signed integers); "date" (date: yyyy-mm-dd); "time" (coordinated universal time: hh:mm:ss.decimal); "utc" (date and time: yyyy-mm-dd hh:mm:ss.decimal).

```
quantum (str|norm|word|name|sint|date|time|utc) "norm"
```

QUICKTIME (notation) Q.2

Notation that should be declared for a QuickTime video file other than a `.mov` file.

```
<!NOTATION QUICKTIME SYSTEM "QUICKTIME">
```

Example:

```
<!ENTITY clip001 SYSTEM "../qt/clip001.qt" NDATA QUICKTIME>
```

The file should be stored in a `qt` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM document. QUICKTIME files may be included in TEDD packages experimentally. Preferred formats will be specified in future TCIF Guidelines.

It is not known yet whether this notation is needed, since QuickTime is not well documented and many variations exist. QuickTime movies in `.mov` files that can be demonstrated to play on an MS Windows PC should be given the notation MOV (M.7).

"Quote", type= — P (P.1), S (S.1), T **(T.1)**

"QuoteAttrib", type= — P (P.1)

R (RCDATA — RTF)

RCDATA (SGML reserved name) R.1

"Replaceable character data." This is a data content type distinguished from CDATA (C.4) by the fact that character entity references are recognized, and distinguished from PCDATA (P.3) by the fact that other markup (particularly subelements) is not recognized. This data type is not declared anywhere in TIM, and is not permitted in XML.

RCverbatim (notation, TIM 1 only)

A notation in which characters were to be treated as RCDATA (R.1).

"ReaderComment", type= — Annote (A.9)

<Ref> R.2

Simple reference, used mainly for in-line references in text to such things as footnotes, tables, and bibliographic citations. Its content could be empty, or a number or mark, or a text phrase. A **Ref** must have an **rid** attribute referencing an element within the document, but using a HyTime **NameLoc** element, this could actually create a link to another document. A **URLRef** (U.4) has a **url** attribute referencing a Uniform Resource Locator, which could be location within the document or anything on the World Wide Web.

The **include** attribute (I.7) can be used to specify what the content of the element should be if the supplied content is inappropriate (for instance, a page number). The **target** attribute can be used to specify the element type or other type of target for the link so that appropriate content can be generated.

```
<!ELEMENT Ref - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef|IndexTerm
          |Pt|Graphic|Object|ExternalText|SimpleList|Equation
          |GraphicalText)*>
<!ATTLIST Ref
          %baseatts;
          %identity;
          %commonatts;
          %textatts;
          rid IDREF #REQUIRED
          target NMTOKEN #IMPLIED
          prefix CDATA #IMPLIED
          infix CDATA #IMPLIED
          suffix CDATA #IMPLIED
          include NMTOKENS #IMPLIED
          HyTime NMTOKEN #FIXED "clink"
          HyNames CDATA #FIXED "linkend rid"
          refrange NMTOKENS "rid X">
```

Values of the **type** attribute have not yet been established. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%commonatts;** (C.19), and **&textatts;** (T.12) for more on this element's attributes.

Comment

The text of a **Ref** or **URLRef**, like the text provided within **Contents** and **Index** elements, may be replaced or discarded by the rendering application, because it may not be valid in the recipient's medium or format. References to page numbers will almost certainly be incorrect, but even references to section numbers, figure numbers, etc., may be wrong, because item numbering may be overridden by the recipient for consistency. (**Contents**, **Index**, **Ref**, and **URLRef** are the only elements whose content may be altered by recipients under normal terms and conditions of document interchange. Numbering and presentation attributes may be changed for any element. And the originator may indicate, as with use of **present="altrep"** or **revstat="strike"**, that presentation is optional.)

Style Notes

Page numbers in cross-references are strongly discouraged. If the document is to be delivered electronically, even a paper copy printed by the recipient is unlikely to have the same pagination as the original paper copy. Cross-references to numbered structural units like **Section**s and **Frame**s will work in all media.

There is a widespread opinion, though not a consensus, that textual cross-references should be self-contained whenever possible, so that they can be discarded (or replaced by iconic hyperlinks) when necessary, without awkwardness. One amenable pattern is space-paren-text-paren:

```
... use PNG format<Ref rid="..."> (P.11)</Ref> for archiving
full-color images ...
```

Ref text that begins with a comma and space works within parens and in most other situations where parens don't work. For automatically generated content in cross-references, variations that include words like "Chapter", "Figure", and "Equation" provide enough choices to fit most circumstances.

"References", type= — Section (S.4)

refrange= R.3

Reference resolution range. Tells HyTime processors how far they must be prepared to look for the element referenced by an IDREF. When the value ends in "X", as it does for all TIM elements, the target element may be in an external document. The definition for **Link** (L.9) is:

```
refrange CDATA #FIXED "#ALL X"
```

For **Ref** (R.2), the definition is:

```
refrange NAMES "rid X"
```

"RegTM", type= — T (T.1)

"Release", type= — T (T.1)

<RelDoc> R.4

Occurs any number of times within **DocID** (D.27) to identify information products that are related to the current document (but are not subdocuments, superdocuments, or documents replaced by the current document). Each instance contains a document number (**DocNo.u** or **DocNo**), and optionally a simple **Title** or complete **TitleGroup**.

```
<!ELEMENT RelDoc - - ((DocNo|DocNo.u),(Title|TitleGroup)?)>
<!ATTLIST RelDoc
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Relevant example:

```
<RelDoc><DocNo><DN.base>TCIF-IPI-95-001</DN.base></
DocNo><Title>The Telecommunications Electronic Document
Delivery (TEDD) Package</Title></RelDoc>
```

Including the COMMON LANGUAGE code for the originating company in the document number is optional and sometimes advisable.

remap= R.5

Remapping information. Contains information needed to convert TIM elements to elements in another markup language. Useful primarily when a TIM document is the result of a conversion from a semantically richer markup. For instance, a Glossary section in a document translated from the DocBook DTD would be tagged **<Section type="Glossary" remap="Glossary" remapto="DocBook">**. Most recipients will want their applications to handle only those remap values that correspond to elements used in their internal DTDs.

```
remap CDATA #IMPLIED
```

Default value: undefined (no remap value provided).

This attribute is defined for all elements that have the **%baseatts;** (B.2).

Style Note

Since many markup sets use similar tag names (TIM and DocBook, for example), use of a **remapto** with every **remap**, to identify the original markup, is recommended. Example:

```
<P remap="FormalPara" remapto="DocBook">
```

These wordings are recommended: "AAP"; "ATA"; "CALS"; "DocBook"; "HTML2.0"; "IBMIDDOC"; "ISO-12083"; "J2008"; "Q". Mappings to specific MS Word style sheets, FrameMaker+SGML EDDs, etc., will be company-specific and so are best identified with a company code.

remap.anchrole= R.6

Since **Link** (L.9) has no specific meaning or direction, remapping attributes are provided to preserve information about more specific kinds of links in other markup that have been converted to TIM **Link**s. This one preserves specific HyTime **anchrole**s (A.7).

```
remap.anchrole NMTOKENS #IMPLIED
```

remap.extra= R.7

Preserves information about values of the HyTime attribute **extra** (E.17) in a more specific kind of link translated to a TIM **Link** (L.9).

```
remap.extra NMTOKENS #IMPLIED
```

remap.intra= R.8

Preserves information about values of the HyTime attribute **intra** (I.15) in a more specific kind of link translated to a TIM **Link** (L.9).

```
remap.intra NMTOKENS #IMPLIED
```

remap.refrange= R.9

Preserves information about values of the HyTime attribute **refrange** (R.3) in a more specific kind of link translated to a TIM **Link** (L.9).

```
remap.refrange CDATA #IMPLIED
```

remapatt= R.10

A mechanism to preserve attributes from another markup that do not exist in TIM. Any number of attributes and their values may be included in the value of the **remapatt**, but they must all use only one of the two kinds of quotation marks for their values, so they can all be enclosed in the other.

```
remapatt CDATA #IMPLIED
```

Example:

```
<Title remapatt="id='title03' pagenum='3'" remapto="DocBook">
```

remaps= R.11

Specifies what values of the **remap** attribute elements should have to be listed in a **Contents** list or an **Index**. See **Contents** (C.22) for more explanation.

```
remaps CDATA #IMPLIED
```

The default value is undefined — that is, values of **remap** should be ignored and all elements that match the remaining specifications should be included.

remapto= R.12

This attribute should be used whenever the **remap** attribute is used, to identify the source encoding (such as DocBook or some other DTD, HTML, etc.) in which the **remap** name is meaningful.

```
remapto CDATA #IMPLIED
```

This attribute is defined for all elements that have the **%baseatts;** (B.2).

REQUIRED (SGML reserved name) R.13

A REQUIRED attribute is one whose value must be explicitly specified in order for the element to be valid. For example, a **Ref** element must have an **rid** value (a pointer to another element) assigned to it or it will not reference anything.

requirement — DistVarLI (D.6), rqmts (R.22)

<Resources> R.14

An holder, new in TIM 2, for all elements whose position in the document is irrelevant: **Link** (L.9), **DataLoc** (D.2), **NameLoc** (N.3), and **TreeLoc** (T.22). In TIM 1, these were inclusion elements, but all inclusion elements have been removed from TIM 2 for XML compatibility.

"restarts", continuation= (C.23)

restartsat= R.15

Specifies a number other than 1 at which numbering of sections, list items, frames, paragraphs, etc., should restart. See **continuation** (C.23).

```
restartsat NUMBER #IMPLIED
```

Default value: should be assumed to be "1" if **continuation** has the value "restarts", undefined otherwise.

The **restartsat** attribute is defined, for the elements that have it, through the entity reference **%numbering;** (N.13).

"Result", type= — LI (L.8)

"revflag", revstat= (R.16)

"ReviewerComment", type= — Annote (A.9)

"revised", revstat= (R.16)

"Revisions", type= — Contents (C.22)

revstat= R.16

Revision Status. Indicates whether an element's contents have been revised. The values "revflag" and "insflag" mean more than that the contents have been revised or inserted: It also means that they should be marked as such, e.g., with change bars as shown here. The value "strike" means that presentation is optional: if presented, the contents should be struck through, ~~like this~~.

```
revstat (norev|revised|revflag|insert|insflag|delete|strike)
#IMPLIED
```

This attribute is defined for all elements that have the **%baseatts;** (B.2). The value is inherited from the parent element; the default is **norev** for the top document element.

revstats= R.17

Specifies what values of the **revstat** attribute (R.16) an element should have to be listed in a **Contents** list or an **Index**. See **Contents** (C.22) for an explanation.

```
revstats CDATA #IMPLIED
```

The default value is undefined — all values of **revstat** should be included.

rid= R.18

Referenced Identifier. Used by **Ref** elements (R.2) to identify the objects they point to. **DBRef** elements use a similar attribute, **dbid**, defined as NAME, for references to external documents.

```
rid IDREF #REQUIRED
```

rids= (TIM 1 only)

IndexEntry (I.10) now uses **Ref**s for its links.

"right", align= (A.4)

"right", presspec= (P.17)

"roman", emph= (E.6)

rotate= R.19

Applies to **Entry** only. A value of "1" means that the content of the cell should be rotated 90 degrees counterclockwise for presentation on an ordinary paper page. The default is "0" (no).

```
rotate NMTOKEN "0"
```

<Row> R.20

A table row. It is contained within a **TBody**, **THead**, or **TFoot** and contains an **Entry** (or **EntryTbl**) for each column (fewer if a **SpanSpec**, S.20, is in effect).

```
<!ELEMENT Row - - (Pt*,(Entry|EntryTbl),(Pt|Entry|EntryTbl)*) >
<!ATTLIST Row
          id ID #IMPLIED
          type NMTOKENS #IMPLIED
          dstype CDATA #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          group NMTOKENS #IMPLIED
          status CDATA #IMPLIED
          version IDREFS #IMPLIED
          revstat (norev|revised|revflag|insert|insflag|delete|strike)
            #IMPLIED
          lang CDATA #IMPLIED
          present (normal|float|alert|altrep) #IMPLIED
          presspec CDATA #IMPLIED
          rowsep NMTOKEN #IMPLIED
          valign (top|middle|bottom) #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Use of the **type** attribute is not supported.

rowsep= R.21

Applies to **Table**, **TGroup**, **ColSpec**, **SpanSpec**, **Row**, **Entry**, and **EntryTbl**. A value of "1" means that a horizontal rule should separate rows, and a value of "0" means not. Values are inherited from higher-level elements. At the row or

entry level, the attribute applies to the bottom separator. The value for the last row is ignored. No default is specified, so the application may have to supply one.

```
rowsep NMTOKEN #IMPLIED
```

"RqmtObj", type= — DistOrdLI (D.6) and DistVarLI **(D.7)**

"RqmtObjs", type= — Contents (C.22) and Index **(I.9)**

%rqmts; R.22

An entity reference defined in TIM to allow an extension for markup of "requirement objects" (requirements and related structures like conditions, objectives, and tests). The extension can be made by putting into the document prologue an entity declaration for **%rqmts;** that points to an actual file (the default definition does not). Example:

```
<!ENTITY % rqmts SYSTEM "../../lib/rqmtobjs.dtd" NDATA SGML>
```

It is assumed that some of the declarations in the file would be for new elements, and the rest would redefine elements already in the TIM DTD, so that they would be allowed to contain the new elements. Since TIM reads these declarations (if they exist) before its own, the new ones will be in effect. There are seven entity references for extensions to TIM 2, and their placement (before all element declarations in the DTD) and order (**%other;**, **%memos;**, **%rqmts;**, **%procs;**, **%eqns;**, **%flows;**, **%tables;**) are significant, since the order is the priority for redefining elements: only the first element and attribute declarations for a given element are used. The order shown reflects the likelihood that declarations in one extension would be redefining the elements in others: the table extension (already provided) is, of course, the least likely to affect others.

By default, **%rqmts;** is a null string, so its occurrence in the DTD does nothing:

```
<!ENTITY % rqmts        "">
%rqmts;
```

RTF (notation) R.23

A text notation containing RTF (Microsoft Rich Text Format) markup.

```
<!NOTATION RTF       SYSTEM "RTF">
```

It may be within a TIM file:

```
<Listing format="RTF">{\rtf1\ansi ....
```

Or it may be in a separate file with a `.rtf` extension, in an `rtf` directory at the same level as the `mtext` directory containing the TIM file(s):

```
<!ENTITY text001 SYSTEM "../rtf/rtf001.rtf" NDATA RTF>

...

<ExternalText entityref="text001">
```

S (S — SYSTEM)

<S> S.1

General structural-grouping (or "block") element. **S** structures can appear within any level of the **Section** hierarchy, but are not themselves indicators of hierarchical structure, only of grouping. Structures can be used (with appropriate **type** or **dstype** values) to represent special constructs like requirement objects, procedures, or any other kind of larger-than-paragraph structural unit. But **S** has only grudgingly been given numeration attributes; a numbered (and, by implication, recurring) structure should instead be tagged as a distributed-list item. **S** cannot contain **#PCDATA** directly; text must be within subelements.

```
<!ELEMENT S - - ((Title|TitleGroup)?,(P|S|Annote|Frame|Danger|Caution
          |Warning|Admon|DistOrdLI|DistVarLI|Table|OrderedList
          |UnorderedList|VariableList|SegmentedList|Listing|Pt|IndexTerm
          |Graphic|Object|Flowchart|ExternalText)*) >
<!ATTLIST S
          %baseatts;
          %identity;
          %numbering;
          %commonatts;>
```

See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), and **%commonatts;** (C.19) for all this element's attributes. Unrecognized values of the **type** attribute should be anticipated (that is, recipients should have a strategy for dealing with them), because **S** is conceived as a kitchen-sink element. Recipients should also be prepared for unexplained, untitled, untyped **S** elements used merely as wrappers to solve authors' structuring or formatting problems.

Style Note

One use of **S** is as a featureless wrapper for a structural grouping within a paragraph, such as a sequence of **Listing**s, so that the presence of PCDATA in the content model of **P** does not cause all line breaks between Listing elements to be rendered as data characters (extra white space or line breaks in the text).

"Safety", type= — Admon (A.2)

scope= S.2

Specifies where to look for elements to include in a contents list or index: All elements listed should be searched for subelements matching the other criteria specified (**elements**, **levels**, **types**, etc.). See **Contents** (C.22) for more.

```
scope NMTOKENS #IMPLIED
```

The default value is undefined, meaning that the whole document is to be searched. "Body" or "Body AppMatter" may be more useful values for most

purposes. The special value "Parent" means the parent element of the current **Contents** or **Index** element, for section-level contents lists and indexes. **Contents** and **Index** elements themselves should not be searched. This does not mean that an Index cannot be listed in a Table of Contents — it will be if the value for **elements** includes "Index". It does mean that subelements of contents lists and indexes cannot be included.

"ScreenImage", type= — Listing (L.12)

"ScreenText", type= — Listing (L.12)

SDATA (SGML reserved name) S.3

"System data." This is a data content type declared for character entity references. It means that once the SGML processing has substituted the declared value for a character entity (e.g., "[amp]" for "&", which represents an ampersand character), it should ignore the substituted characters and leave them to be interpreted by the "system."

<Secondary> (TIM 1 only)

Replaced by a nested **IndexEntry** (I.10).

<Section> S.4

Sections (any kind and any level) within the major subdivisions of a document. Sections may contain paras, **S** structures, lists, admonishments, annotations, frames, tables, external objects, listings, or subsections. They may have titles, and they may have labels (most likely, numbers). The TIM DTD does not define special kinds of sections, such as prefaces, appendixes, or glossaries: Any of those would be a **Section** with an appropriate value of **type** (T.23).

```
<!ELEMENT Section - - ((Title|TitleGroup)?,(Contents|Index|P|S|Annote
          |Frame|Danger|Caution|Warning|Admon|DistOrdLI|DistVarLI|Table
          |OrderedList|UnorderedList|VariableList|SegmentedList|Listing
          |Pt|IndexTerm|Graphic|Object|Flowchart|ExternalText
          |Section)*) >
<!ATTLIST Section
          %baseatts;
          %identity;
          alerts IDREFS #IMPLIED
          altreps IDREFS #IMPLIED >
```

Values of the **type** attribute that should be recognized: "Abstract", "Disclaimer", "TradeAcknowl" (trademarks), "History" (of revisions), "Changes", "Audience", "Preface", "ExecSum", "About", "How-To-Use" (these are normally within **FrontMatter**); "Part", "Chapter", (normally within **Body**); "Appendix", "Glossary", "References", "Bibliography" (normally within **AppMatter**). More-specific values should be anticipated. See the entries for the

entities **%baseatts;** (B.2) and **%identity;** (I.3) for more on this element's attributes.

Sections should, by default, inherit numbering properties (that is, the values of attributes, not the section numbers themselves) from the most recent **Section** of the same hierarchical level, even if a higher-level **Section** has begun since. If no **Section** of the same hierarchical level has occurred, numbering properties should be inherited from the most recent **Section** at the next higher level. Thus the first **Section** within a **TDoc** will set default numbering properties (including inheritance) for all other **Section**s, and the first **Section** at each lower level will reset them for that level and lower. In the following example, the **label** shows what the number should be:

```
<TDoc><Section numeration=arabic inheritnum=inherit suffix="."
label="1.">
<Section infix="." suffix="" label="1.1">
...
</Section>
</Section>
<Section label="2.">
<Section label="2.1">
<Section label="2.1.1">
```

Comment

The ambiguity of leaving **Section** levels implicit is offset by the flexibility it provides. Moving a section to another document or another place in the same document would often require retagging if section levels were explicit ("Sect1," "Sect2," etc.)

Comment

The content models of **Section** and its parent elements (**Body**, etc.) allow arbitrary elements to occur between or after sections as well as before them. Elements occurring before the sections are the common case of introductory material. But elements occurring between or after sections have virtually no analog in existing paper documents, and most rendering systems could not be expected to have a way of dealing with them (that is, most would not have a way of indicating the end of a section except by the start of a new one). Therefore such element sequences should usually be avoided. (The content model allows them since we expect they might be useful in documents created for hypertext.)

<See> (TIM 1 only)

Dropped as unnecessary. Use a **Ref** within **IndexEntry** (I.10).

<SeeAlso> (TIM 1 only)

Dropped as unnecessary. Use a **Ref** within **IndexEntry** (I.10).

<SegHead> (TIM 1 only)

Replaced by **ListHead** (L.11).

<SegLI> S.5

Segmented-list item, essentially a row of a columnized list. See **SegmentedList** (S.7). It should contain the number of ordinary list items (**LI**s) specified by the **segments** attribute of the **SegmentedList** or **DistSegLI**. Any numbering of the **SegLI** items in a **SegmentedList** is specified by the enclosing **SegListGrp**.

```
<!ELEMENT SegLI - - (Pt*,LI,(Pt|LI)*) >
<!ATTLIST SegLI
          %baseatts;
          %identity;
          %commonatts;>
```

Values of the **type** attribute have not yet been established. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), and **%commonatts;** (C.19) for all this element's attributes.

Comment

Be careful to put the specified number of **LI**'s in each **SegLI**. Parsers will check what subelements are within each element but not how many.

<SegListGrp> S.6

A section of a **SegmentedList** (S.7), analogous to **TGroup** within a table. It may have different headings for the mark and segments of the list items within it, or it may specify numbering for the items within it. The numbering specified by the **SegListGrp**'s attributes belongs to the **SegLI**s (the rows), not the **LI**s within them.

```
<!ELEMENT SegListGrp - - (ListHead?,(Annote|Danger|Caution|Warning|Admon
          |SegLI|Pt)*) >
<!ATTLIST (SegListGrp|OrderedList|OrdListGrp|VariableList|VarListGrp)
          %baseatts;
          %identity;
          %numbering;
          %commonatts; >
```

Use of the **type** attribute is not supported. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), and **%commonatts;** (C.19) for all this element's attributes.

Comment

If **Titles** are specified (in the optional **ListHead**), there should be as many as there are "segments" in the list, plus another if there are labels on the items. (The element content model can't predict this number, so parsers will not enforce this rule.)

<SegmentedList> S.7

A segmented list, one in which each item (row) has segments in a columnar alignment. Such lists are common in telecommunications practices.

The required **segments** attribute specifies the number of in-line items (columns). Each **SegLI** (row) should have that number of items (**LI** elements). Headings for the segments are specified in a sequence of **Titles** wrapped in a **ListHead** (L.11). There may also be a **Title** over the list labels.

A **SegmentedList** is made up of one or more sections, **SegListGrps** (S.6), which define or redefine the list-item numbering and the titles over columns of the list. The **SegmentedList**'s own numbering attributes apply to the list as a whole, not the list items. It is reasonable to number the list as if it were a table, since it may be rendered as one.

```
<!ELEMENT SegmentedList - - ((Title|TitleGroup)?,(SegListGrp+)) >
<!ATTLIST SegmentedList
          %baseatts;
          %identity;
          %commonatts;
          colsep NMTOKEN #IMPLIED
          frame (top|bottom|topbot|all|sides|none) #IMPLIED
          rowsep NMTOKEN #IMPLIED
          liststyle NMTOKEN #IMPLIED
          segments NMTOKEN #REQUIRED
          segwidths CDATA #IMPLIED >
```

Value of the **type** attribute that should be recognized: "Procedure". See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), and **%commonatts;** (C.19) for more on this element's attributes.

Comment

Tables with numbered rows are better represented in TIM as segmented lists than as CALS tables. Any list in a CALS table must be entirely enclosed in one table cell; no list or other structure can span cells or rows of a table.

Comment

The best way to present a segmented list will vary with the application. In some, they may be "snaking columns"; in others, they may be treated as tables, with

SegListGrp recognized as the element corresponding to **TGroup**, **ListHead** and **Title** as heading row and cell, and **SegLI** and **LI** as body row and cell.

Comment

A segmented list formatted this way —

Step	Action	Result
1	Select **File > New**	"New File" dialog box opens.
2	Type the new file name and hit [Enter]	Your data are saved to the new file.

....

— could be marked up this way —

```
<SegmentedList type="Procedure" segments=2 widths="3* 2*">
<SegListGrp numeration=arabic>
<ListHead>
<Title>Step</Title><Title>Action</Title><Title>Result</Title>
</ListHead>
<SegLI>
<LI><P>Select <T type="MenuItem">File > New</T>.</P></LI>
<LI><P>"New File" dialog box opens.</P></LI>
</SegLI> ...
```

— or this way —

```
<SegmentedList type="Procedure" segments=3 widths="1* 3* 2*">
<SegListGrp numeration=none>
<ListHead>
<Title>Step</Title><Title>Action</Title><Title>Result</Title>
</ListHead>
<SegLI>
<LI emph="bold"><P>1</P></LI>
<LI><P>Select <T type="MenuItem">File > New</T>.</P></LI>
<LI><P>"New File" dialog box opens.</P></LI>
</SegLI> ...
```

segments= S.8

Number of items (i.e., segments, columns) in a **SegLI** (row) of a segmented list (S.7):

```
segments NMTOKEN #REQUIRED
```

An error occurs in parsing if the number is not specified.

segwidths= S.9

Specifies the widths of the segments (columns) in a **SegLI/DistSegLI** (i.e., row) of a **SegmentedList** (S.7). Following the CALS Table **colwidth** design, each width may be an absolute or relative measurement, e.g., "3p" for 3 picas, "36pt" for 36 points, or "1*" for one unit, "2*" for two units, twice as wide as 1*. Each measurement should start with a numeral and should not contain spaces — fractional units must be expressed as "0.125in".

```
segwidths CDATA #IMPLIED >
```

Units of measure that should be understood (the same as for **colwidth**): "mm" (millimeter); "cm" (centimeter); "in" (inch); "p" (pica, 1/6 in), "pt" (point, 1/72 in); "dd" (didot, 1/67.431 in); "cc" (cicero, 1/5.619 in, 12 didots); a number with neither specified units nor "*" should be interpreted as pixels. Use lower case. Two units of measure cannot be combined in one value (e.g., "3p6" should be "3.5p" or "42pt").

The default value is equivalent to "1*".

separator= S.10

How items before the last one of a **SimpleList** with **arrange="inline"** should be separated. See the example at **SimpleList** (S.15).

```
separator CDATA #IMPLIED
```

There is no default, although ", " (comma, space) could be one.

"Service", type= — Admon (A.2)

"Set", type= — SuperTitle (S.32)

SGML (notation) S.11

A text notation containing SGML markup other than TIM.

```
<!NOTATION SGML      PUBLIC "ISO 8879:1986//NOTATION
          Standard Generalized Markup Language//EN">
```

It may be within a TIM file:

```
<Listing format="SGML"><!DOCTYPE Book ....
```

Or it may be in a separate file with an `.sgm` or DTD-specific extension, in the same `mtext` directory/folder containing the TIM file(s):

```
<!ENTITY sgml001 SYSTEM "sgml001.sgm" NDATA SGML>

...

<ExternalText entityref="sgml001">
```

<ShortTitle> S.12

An optional short form of a title (of anything that can have a title, except a list item). May be provided for running headers of pages, tables, etc. Occurs only within a **`TitleGroup`** (T.21).

```
<!ELEMENT ShortTitle - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef
          |IndexTerm|Pt|Graphic|Object|ExternalText|SimpleList|Equation
          |GraphicalText)* >
<!ATTLIST ShortTitle
          %baseatts;
          %commonatts;
          %textatts; >
```

Use of the **`type`** attribute is not supported. See the entries for the entities **`%baseatts;`** (B.2), **`%commonatts;`** (C.19), and **`&textatts;`** (T.12) for all this element's attributes.

shownum= S.13

An attribute for numbered elements specifying which, if not all, elements should have their numbers visible. For elements that are individually unnumbered, like **`Section`**, **`DistOrdLI`**, and **`Frame`**, the values are "1" (show the number, the default) and "0" (do not show the number, which is not the same as no number because it increments the counter for the next such element). For elements whose numbering attributes specify the numbering of subelements, like **`OrderedList`** or **`OrdListGrp`**, **`shownum`**'s value may be one to three numbers counting off the first, second, and third intervals between items on which numbers are shown. The last number specifies the interval that continues through the list. Examples: **`shownum="2"`** means show the number on items 2, 4, 6 ...; **`shownum="1 4 5"`** means show the number on items 1, 5, 10, 15 ...;

```
shownum NMTOKEN #IMPLIED
```

The default is always "1", meaning number every item. The **`shownum`** attribute is defined, for the elements that have it, through the entity reference **`%numbering;`** (N.13).

Comment

Document originators shouldn't expect rendering applications to handle any of the infinite number of possible values for this attribute. The following are

probably the only reasonable values: "1", "2", "3", "5", "10", "1 2", "1 2 3", "1 4 5", "1 9 10".

"Sidebar", type= — Frame (F.8)

"sides", frame= (F.9)

<SimpleLI> S.14

An item of a **SimpleList** (S.15).

```
<!ELEMENT SimpleLI - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef|Pt
          |IndexTerm|Graphic|Object|ExternalText|Equation
          |GraphicalText)* >
<!ATTLIST SimpleLI
          %baseatts;
          %identity;
          %commonatts;
          %textatts; >
```

Values of the **type** attribute have not yet been established. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%commonatts;** (C.19), and **&textatts;** (T.12) for all this element's attributes.

<SimpleList> S.15

A list whose items are simple enough to be handled by the **T** (text phrase) content model of the **SimpleLI** element (S.14). SimpleList has four attributes not used elsewhere. **Arrange**, which can have the values **inline**, **vert**, and **horiz**, specifies the preferred way of arranging the list terms. When **arrange="horiz"** or **arrange="vert"**, the second attribute, **columns**, is meaningful: how many columns the arrangement should have. Examples:

```
 ... uses the keywords <SimpleList arrange="inline"
separator=", " lastsep=", and "><ListTerm>horiz</
ListTerm><ListTerm>inline</ListTerm><ListTerm>vert</ListTerm></
SimpleList>.
```

should yield:

. . . uses the keywords horiz, inline, and vert.

while

```
... uses the keywords <SimpleList arrange="vert" columns=2> ...
```

should yield:

. . . uses the keywords

horiz vert.

inline

and

```
... uses the keywords <SimpleList arrange="horiz" columns=2>
...
```

should yield:

. . . uses the keywords

horiz inline

vert.

The separating ", " between in-line items can be specified with the **separator** attribute, and **lastsep** can be used to specify ", and " as the last separator. If they are not specified, the rendering application would have to supply default separators; it always has to supply the indent and column spacing for vertical and horizontal items.

```
<!ELEMENT SimpleList - - (SimpleLI+) >
<!ATTLIST SimpleList
          %baseatts;
          %identity;
          %numbering;
          %commonatts;
          separator CDATA #IMPLIED
          lastsep CDATA #IMPLIED
          arrange (inline|vert|horiz) #IMPLIED
          columns NMTOKEN #IMPLIED >
```

Values of the **type** attribute have not yet been established. The default for **arrange** is "inline". See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), and **%commonatts;** (C.19) for more on this element's attributes.

"sint", quantum= (Q.1)

sortas= S.16

Provides a way to alphabetize numerals and other nonalphabetic characters in an element collected into an **Index** (I.9) as if they were spelled out:

```
... <IndexTerm rids="..." sortas="pound sign">#  (Pound sign)</
IndexTerm> ...
```

would place the index entry here:

POTS

(Pound sign)

Power Supplies

The default value is undefined — sorting is based on the content of the element as is.

```
sortas CDATA #IMPLIED
```

This attribute is defined for all elements that have the **%textatts;** (T.12).

sortorder= S.17

Specifies, for **Index** (I.9), where nonalphabetic characters should sort relative to alphabetic characters and each other. Classes of characters as well as individual characters can be specified, with the spec for a character (or the entity representing it, like **•** for a bullet) taking precedence over the spec for the character's class. **Sortorder** cannot be used to specify complex algorithms such as sorting numbers as if they were spelled out — for that, use the **sortas** attribute (S.16) on the individual **IndexEntry**s (or other elements).

```
sortorder CDATA #IMPLIED
```

Values that should be recognized, besides individual characters and entities: "whitespace" (spaces, tabs, line breaks); "upperfirst" (which means AaáBb, not ABabá); "lowerfirst"; "mixedcase" (mixed-case, A=a=á and b=B); "greek" (any non-latin alphabetics); "numeral"; "punctuation"; "accent"; "symbol" and "ignore" (ignore anything listed after this term). It should be realized that not all rendering applications will be able to match all possible specifications.

Comment

There is no standard alphanumeric order, but it has become common for spaces to be given lowest value (so "low value" would appear before "lowest value"), then symbols, then numerals, then letters (upper and lower case equivalent), and for punctuation and accent marks to be ignored. Example: ore smelters, Oregon, Ore-Ida, O'Reilly, ores. The **sortorder** value to specify this default is:

```
sortorder="whitespace symbol numeral mixedcase ignore
punctuation accent"
```

(Put "greek" where you like.)

<Span> (TIM 1 only)

Its job is now done by **Pt** (P.23).

spanend= S.18

An attribute of **Pt** (P.23) that points to the element at the end of a virtual structure (a range of content that cannot be enclosed in a single element because of the existing SGML tree) that represents an element in an alternative tree.

```
spanend IDREF #IMPLIED
```

spanname= S.19

Applies to **SpanSpec**, **Entry**, and **EntryTbl**. Provides a name for a column-spanning specification, which can be used for further spanning (by specifying a **namest** or **nameend** in an **Entry** that is a **spanname** instead of **colname**). The definition for **SpanSpec** is:

```
          spanname NMTOKEN #REQUIRED
```

The definition for **Entry** and **EntryTbl** is:

```
          spanname NMTOKEN #IMPLIED
```

<SpanSpec> S.20

May appear within a **TGroup** (T.14) to specify which columns — they must be named in **ColSpec Colname** attributes (C.16) — will be spanned by cells in the rows within the **TGroup**.

```
<!ELEMENT SpanSpec - O EMPTY >
<!ATTLIST SpanSpec
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          align (left|right|center|justify|char) "center"
          char CDATA #IMPLIED
          charoff NMTOKEN #IMPLIED
          colsep NMTOKEN #IMPLIED
          nameend NMTOKEN #REQUIRED
          namest NMTOKEN #REQUIRED
          rowsep NMTOKEN #IMPLIED
          spanname NMTOKEN #REQUIRED
          TIM2 NMTOKEN #IMPLIED >
```

"splitarabic", splitnum= (S.21)

"splitlalpha", splitnum= (S.21)

"splitlroman", splitnum= (S.21)

splitnum= S.21

Specifies that a numbered item should be numbered as one of two or more items with the same base number. Huh? Well, for instance, in a list of five items with **numeration=arabic**, if the third and fourth items have the attribute **splitnum=splitlalpha**, then numbering would be 1, 2, 3a, 3b, 4. The alternative of making the same items a sublist would yield 1, 2, 2a, 2b, 3 (with **inheritnum=inherit**), which is sometimes not what's intended.

There is no provision for a separator between the base and added numbers (**infix** is only for numbers concatenated with **inheritnum**), so it is necessary to choose a value for **splitnum** that complements the value of **numeration** (**numeration=arabic splitnum=splitarabic** yields 1, 2, 31, 32, 4)

If numbering changes within a block list (**OrderedList**, **VariableList**, etc.), list sections must be used, because list numbering is defined at the list-section level, not the item level: see **OrdListGrp** (O.5), **SegListGrp** (S.6), and **VarListGrp** (V.4).

```
          splitnum
(nosplit|splitarabic|splitualpha|splitlalpha|spliturоman
          |splitlroman|splitoutline) #IMPLIED
```

The default value is "nosplit." The **splitnum** attribute is defined, for the elements that have it, through the entity reference **%numbering;** (N.13).

"splitoutline", splitnum= (S.21)

"splitlualpha", splitnum= (S.21)

"splitluroman", splitnum= (S.21)

<Status> S.22

Occurs one or more times within **DocID** (D.27) to provide information about the status of a document (**status** is also an attribute, S.23).

```
<!ELEMENT Status    - -  (#PCDATA)* >
<!ATTLIST Status
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

This attribute is defined for all elements that have the **%baseatts;** (B.2).

Specific uses and specific forms for the contents of **Status** elements are left to the discretion of the document originator.

Examples:

```
<Status>Draft</Status>

<Status>Revised 93/08/03</Status>

<Status>Information contained in this document does not apply
beyond 1999-12-31.</Status>
```

status= S.23

(**Status** is also an element, S.23) Typically this attribute would be used to differentiate among concurrent versions of an element or track its revision history (for successive versions, the **version** attribute, V.8, is more appropriate). Such things are now typically specified only at the **TDoc** (T.9) level, but they could be tracked at lower levels. Examples:

```
<Section status="Ameritech BellSouth">

<P status="Model400">
```

The exact uses and formatting of **status** are left to the discretion of the document originator. The values may or may not be relevant to the recipient, who is free to assign different, locally meaningful values.

```
status CDATA #IMPLIED
```

Default value: undefined for **TDoc**, inherited by each lower-level element from its parent element (that is, any specification of **status** for an element applies to all of its subelements until overridden).

statuses= S.24

Specifies what values of the **status** attribute an element should have to be listed in a **Contents** list or an **Index**. See **Contents** (C.22) for more.

```
statuses CDATA #IMPLIED
```

The default value is undefined — that is, all values of **status** should be included.

"Step", type= — DistOrdLI (D.6) and LI (L.8)

"str", quantum= (Q.1)

"strike", revstat= (R.16)

<Sub> S.25

Subscript, which can occur within any level of text and can contain anything allowed for **T** phrase structures, including another level of subscripting.

```
<!ELEMENT Sub - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef|IndexTerm|Pt
          |Graphic|Object|ExternalText|SimpleList|Equation
          |GraphicalText)*>
<!ATTLIST Sub
          %baseatts;
          id ID #IMPLIED
          emph (ital|bold|boldital|roman|und|dblund|color|sys) #IMPLIED
          %textatts; >
```

Values of **type** have not yet been established. See the entries for **%baseatts;** (B.2) and **&textatts;** (T.12) for more on this element's attributes.

<SubDoc> S.26

Occurs any number of times within **DocID** (D.27) to identify information products that are parts of the current document. Each instance contains a document number (**DocNo.u** or **DocNo**), and optionally a **Title** or **TitleGroup**.

```
<!ELEMENT SubDoc - - ((DocNo|DocNo.u),(Title|TitleGroup)?)>
<!ATTLIST SubDoc
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
```

```
remapatt CDATA #IMPLIED
remapto CDATA #IMPLIED
TIM2 NMTOKEN #IMPLIED >
```

"Subdoc", type= — Subtitle (S.27)

"Subject", type= — Index (I.9) and IndexTerm (I.11)

<SubTitle> S.27

May occur one or more times within a **TitleGroup** (T.21), to specify a subtitle of the main title (of anything that can have a title, except list items).

```
<!ELEMENT SubTitle - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef
          |IndexTerm|Pt|Graphic|Object|ExternalText|SimpleList|Equation
          |GraphicalText)* >
<!ATTLIST SubTitle
          %baseatts;
          %commonatts;
          %textatts; >
```

Values of **type** that should be recognized: "DocType"; "Subdoc" (component document); "Informal". See the entries for the entities **%baseatts;** (B.2), **%commonatts;** (C.19), and **&textatts;** (T.12) for all this element's attributes.

suffix= S.28

An optional suffix for the number of a section, list item, frame, paragraph, etc. It is most often a period (".").

Any separating spaces or tabs should be included at the end of a **prefix**, the beginning of a **suffix**, and both ends of an **infix**, since they will not be assumed by the rendering application. Separators should not be included at the beginning of **prefix**, the end of **suffix**, or either end of **label**. These are supplied by the rendering application.

```
suffix CDATA #IMPLIED
```

The **suffix** attribute is defined, for the elements that have it, through the entity reference **%numbering;** (N.13). The default is undefined — a rendering application must be prepared to choose prefixes and suffixes for numbered elements. Since some rendering applications can't distinguish a null attribute value, "", from an undefined one, the special value "nosuffix" can be used as an explicit "", meaning use no suffix rather than choosing a different default.

SUNRASTER (notation) S.29

Notation that should be declared for a `filename.sun` or `filename.ras` SunRaster bitmap graphic file.

```
<!NOTATION SUNRASTER PUBLIC "+//ISBN 0-7923-9432-1::Graphic Notation//
NOTATION
          Sun Microsystems raster//EN">
```

Example:

```
<!ENTITY fig001 SYSTEM "../sun/fig001.sun" NDATA SUNRASTER>
```

The file should be stored in a `sun` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM document. SUNRASTER files should not be included in TEDD packages that follow all TCIF Guidelines. Currently the TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG (P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18).

<Sup> S.30

Superscript, which can occur within any level of text and can contain anything allowed in **T** phrase structures, including another level of superscripting.

```
<!ELEMENT Sup - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef|IndexTerm|Pt
          |Graphic|Object|ExternalText|SimpleList|Equation
          |GraphicalText)*>
<!ATTLIST Sup
          %baseatts;
          id ID #IMPLIED
          emph (ital|bold|boldital|roman|und|dblund|color|sys) #IMPLIED
          %textatts; >
```

Values of the **type** attribute have not yet been established. See the entries for the entities **%baseatts;** (B.2) and **&textatts;** (T.12) for more on this element's attributes.

<SuperDoc> S.31

Occurs any number of times within **DocID** to identify an information product of which the current document is a part. Each instance contains a document number (**DocNo.u** or **DocNo**), and optionally a simple **Title** or complete **TitleGroup**.

```
<!ELEMENT SuperDoc - - ((DocNo|DocNo.u),(Title|TitleGroup)?)>
<!ATTLIST SuperDoc
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Example:

```
<DocID>

...

<DocNo><DN.base>GR-828-CORE</DN.base><DN.iss>1</DN.iss></DocNo>
```

```
<DocDate>88/11</DocDate>

<TitleGroup><SuperTitle>Operations Technology Generic
Requirements (OTGR)</SuperTitle><Title>Generic Operations
Interfaces - OSI Communications Architecture</Title></
TitleGroup>

<SuperDoc><DocNo><DN.base>FR-439</DN.base><DN.iss>2</DN.iss></
DocNo><Title>Operations Technology Generic Requirements</
Title></SuperDoc>
```

"Superdoc", type= — Supertitle (S.32)

<SuperTitle> S.32

May occur one or more times within a **`TitleGroup`** (T.21), to specify a supertitle of the main title (of anything that can have a title, except a list item). A supertitle might specify a set of documents to which a given **`TDoc`** belongs, or might repeat a higher-level section title.

```
<!ELEMENT SuperTitle - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef
          |IndexTerm|Pt|Graphic|Object|ExternalText|SimpleList|Equation
          |GraphicalText)* >
<!ATTLIST SuperTitle
          %baseatts;
          %commonatts;
          %textatts; >
```

Values of **`type`** that should be recognized: "Set"; "Superdoc" (parent document); "Informal". See the entries for the entities **`%baseatts;`** (B.2), **`%commonatts;`** (C.19), and **`&textatts;`** (T.12) for all this element's attributes.

"SvcMark", type= — T (T.1)

<Symbol> S.33

A text symbol; that is, a character in a font that does not contain the usual Latin-alphabet characters.

```
<!ELEMENT Symbol - - (#PCDATA)>
<!ATTLIST Symbol
          %baseatts;
          id ID #IMPLIED
          emph (ital|bold|boldital|roman|und|dblund|color|sys) #IMPLIED
          %textatts;>
```

Values of **`type`** should identify the font needed to present the symbol. Values that should be expected: "Symbol", "Dingbats", "Keycaps", "KeycapExtras", "Buttons", "ButtonsLeft", and "ButtonsRight" (all but the first of these refer to free fonts available from TCIF). See the entries for the entities **`%baseatts;`** (B.2) and **`&textatts;`** (T.12) for more on this element's attributes.

Comment

The default rendering for keycaps is likely to be to enclose each key name in square brackets: "Press [1] followed by [Enter]. For help, press [F1]." Graphical representations or special fonts such as the public-domain KeyCapsTCIF font may be used if available. (You have that font if you are viewing this document on-line and you see rounded boxes around these characters: 🄰🄱🄲.)

"symbol", numeration= (N.15)

symbolseq= S.34

If **numeration** is set to "symbol", as it might be for traditional marking of footnotes, **symbolseq** should be used to specify the sequence of symbols used. Entity references will be needed for non-ASCII characters like daggers. If numbering is not restarted (with **continuation=restarts**) after the end of the specified sequence, the rendering application should repeat the sequence with doubled characters, then tripled, *ad infinitum*.

```
symbolseq CDATA #IMPLIED
```

The **symbolseq** attribute is defined, for the elements that have it, through the entity reference **%numbering;** (N.13). Example: to get the sequence *, †, ‡, §, **, ††, ‡‡, §§, ***, etc., use

```
symbolseq="* &dagger; &Dagger; &sect;"
```

"sys", emph= (E.6)

"SysParam", type= — ListTerm (L.14)

SYSTEM (SGML reserved name) S.35

This contrasts with PUBLIC (P.24), and says that an identifier is to be interpreted by the system (on its own) rather than by looking up its meaning in a catalog of public identifiers. Single-use entities, like graphic files, are generally given SYSTEM identifiers, which either are or can be converted to pathnames that locate a file. Reused entities, like DTDs, are usually given PUBLIC identifiers. This is the way most TIM files begin:

```
<!DOCTYPE TDoc PUBLIC "-//USA/TCIF//DTD TIM-1//EN" [
<!ENTITY fig001 SYSTEM "../tiff/fig001.tif" NDATA TIFF>
....
```

T (T — types)

<T> T.1

Text element. The element for phrase structures (that is, structures occurring within running text). The kind of structure can be indicated by an appropriate **type**:

```
... see <T type="DocID">IPI-4</T> for more on TEDD.
```

Ts can be nested.

```
<!ELEMENT T - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef|IndexTerm|Pt
          |Graphic|Object|ExternalText|SimpleList|Equation
          |GraphicalText)* >
<!ATTLIST T
          %baseatts;
          %identity;
          %commonatts;
          %textatts;>
```

Many values of the **type** attribute have been established: For bibliographies: "ArticleTitle"; "Author"; "DocID"; "DocTitle"; "Editor"; "Issue"; "Page"; "PubDate"; "Publisher"; "Volume". For computer interactions: "Attribute"; "Value"; "Parameter"; 'Code"; "Command"; "Prompt"; "Input"; "Output"; "Fieldname"; "Filename"; "MenuItem". Other: "Acronym"; "Citation"; "Copyrt"; "Date"; "Definition"; "EMail", "ErrorMessage"; "Fault"; "FAX"; "GlossTerm"; "ID"; "Label"; "MathConstant"; "MathFunction"; "MathVariable"; "Name"; "PartNum"; "Phone"; "ProductName"; "Quote"; "RegTM"; "Release"; "SvcMark"; "Tool"; "TrdMark"; "ConstantWidth". **T** should not be used for URLs; use URLRef instead so that a live link will be created. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%commonatts;** (C.19), and **&textatts;** (T.12) for all this element's attributes.

Style Note

T is for text phrases that are marked for semantic reasons — that is, they're something someone might be looking for, like error codes, trademarks, or part numbers. If a word or phrase is made typographically distinct just so it will stand out, use Emph instead.

If the document contains:	Then the recommended markup is:
This is a *Memory Fault* error.	<P>This is a <T type="ErrorName">Memory Fault</T> error.</P>
This is a *serious* error.	<P>This is a <Emph emph="ital">serious</Emph> error.</P>

<Table> T.2

A table. Tables cannot be nested, but effectively one level of nesting can be achieved with the **EntryTbl** element (E.11). Tables do not directly contain or describe cells, rows, or columns — **TGroup** (T.14) and **THead**, **TBody**, and **TFoot** are the intermediate elements.

```
<!ELEMENT Table - - ((Title|TitleGroup)?,TGroup+) >
<!ATTLIST Table
          type NMTOKENS #IMPLIED
          dstype CDATA #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          group NMTOKENS #IMPLIED
          status CDATA #IMPLIED
          version IDREFS #IMPLIED
          revstat (norev|revised|revflag|insert|insflag|delete|strike)
            #IMPLIED
          lang CDATA #IMPLIED
          present (normal|float|alert|altrep) #IMPLIED
          presspec CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED
          id ID #IMPLIED
          label CDATA #IMPLIED
          keywords CDATA #IMPLIED
          numeration (nonum|arabic|outline|upperalpha|loweralpha
            |upperroman|lowerroman|symbol) #IMPLIED
          prefix CDATA #IMPLIED
          infix CDATA #IMPLIED
          suffix CDATA #IMPLIED
          inheritnum (inherit|inherit0|inherit1|inherit2
            |inherit3|noinherit) #IMPLIED
          inheritfrom NMTOKENS #IMPLIED
          splitnum (nosplit|splitarabic|splitualpha|splitlalpha
            |spliturоman|splitlroman|splitoutline) #IMPLIED
          continuation (continues|restarts|holds) #IMPLIED
          restartsat NMTOKEN #IMPLIED
          increment NMTOKEN #IMPLIED
          shownum NMTOKEN #IMPLIED
          symbolseq CDATA #IMPLIED
          emph (ital|bold|boldital|roman|und|dblund|color|sys) #IMPLIED
          alerts IDREFS #IMPLIED
          altreps IDREFS #IMPLIED
          colsep NMTOKEN #IMPLIED
          frame (top|bottom|topbot|all|sides|none) #IMPLIED
          orient (port|land) #IMPLIED
          pgwide NMTOKEN #IMPLIED
          rowsep NMTOKEN #IMPLIED
          tabstyle NMTOKEN #IMPLIED>
```

No values have been established for the **type** attribute. Use **tabstyle** or **presspec**, not **type**, to describe differences in table layout; **type** should be for semantic types. This is the markup for the table in the entry for **T** (T.1), with the **Emph** and **T** elements in the cells ignored:

```
<Table id=TCIF-IPI-95-004-TABLE-9>

<TGroup Align=Left Cols=2>

<ColSpec Colwidth="1.875in">

<ColSpec Colwidth="3.375in">

<THead><Row>

<Entry><P remap='TableHead'>If the document contains:</P></
Entry>

<Entry><P remap='TableHead'>Then the likely markup is:</P></
Entry>

</Row></THead>

<TBody><Row>

<Entry><P remap='TableText'>This is a Memory Fault error.</P></
Entry>

<Entry><P remap='TableText'>&lt;P>This is a &lt;T
type=”ErrorName”>Memory Fault&lt;/T> error.&lt;/P></P></Entry>

</Row>

<Row>

<Entry><P remap='TableText'>This is a serious error.</P></
Entry>

<Entry><P remap='TableText'><P>This is a &lt;Emph
emph=”ital”>serious&lt;/Emph> error.&lt;/P></P></Entry>

</Row>

</TBody>

</TGroup>

</Table>
```

Style Note

Nothing prevents table cells from containing equations, graphics, or arbitrarily complex **S** structures, but the practical problems of rendering such constructs on the page or screen should be a deterrent: generally it is safer to refer to such things in the table but display them separately.

"Table", type= — Frame (F.8)

"TableList", type= — DistOrdLI (D.6)

%tables; T.3

An entity reference defined in TIM to read in the element declarations for table markup. This markup, borrowed almost directly from the CALS Table DTD, was and still is considered the first "extension" of the TIM DTD. Its existence as a separate file leaves open the possibility that a different table model could be substituted.

```
<!ENTITY % tables PUBLIC "-//USA/TCIF//ELEMENTS Modified CALS Tables 2//
EN">
%tables;
```

The PUBLIC identifier in quotes above should be resolved to the file `calstabl2.dtd`, which should accompany the `tim2.dtd` file.

There are seven entity references for extensions to TIM 2, and their placement (before all element declarations in the DTD) and order (**`%other;`**, **`%memos;`**, **`%rqmts;`**, **`%procs;`**, **`%eqns;`**, **`%flows;`**, **`%tables;`**) are significant, since the order is the priority for redefining elements: only the first element and attribute declarations for a given element are used. The order shown reflects the likelihood that declarations in one extension would be redefining the elements in others: the table extension (already provided) is, of course, the least likely to affect others.

"Tables", type= — Contents (C.22)

tabstyle= T.4

Table style name. Applies to **`Table`** only. Specifies a style defined externally, presumably in a FOSI, from which a particular table can inherit formatting defaults. The default value is undefined.

```
tabstyle NMTOKEN #IMPLIED
```

target= T.5

Can be used to specify what kind of element is the target of a **`Ref`** or **`URLRef`**. This can be helpful in generating or regenerating the textual content of the cross-reference.

```
target NMTOKEN #IMPLIED
```

TBL (notation) T.6

A text notation containing **`troff tbl`** markup.

```
<!NOTATION TBL       SYSTEM "TBL">
```

It may be within a TIM file:

```
<Listing format="TBL">.TS ....
```

Or it may be in a separate file with a `.tbl` extension, in a `troff` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s):

```
<!ENTITY tbl001 SYSTEM "../troff/tbl001.tbl" NDATA TBL>

...

<ExternalText entityref="tbl001">
```

The TBL format may be included in TEDD packages experimentally. Preferred formats will be specified in future TCIF Guidelines.

<TBody> T.7

Occurs exactly once within each **`TGroup`** (T.14) of a table and contains the rows of the body of that **`TGroup`**; there may also be header rows in a **`THead`** (T.16) and footer rows in a **`TFoot`** (T.13). Column specifications are given in the **`TGroup`**.

```
<!ELEMENT TBody - - (Pt*,Row,(Pt|Row)*) >
<!ATTLIST TBody
          id ID #IMPLIED
          type NMTOKENS #IMPLIED
          dstype CDATA #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          group NMTOKENS #IMPLIED
          status CDATA #IMPLIED
          version IDREFS #IMPLIED
          revstat (norev|revised|revflag|insert|insflag|delete|strike)
            #IMPLIED
          lang CDATA #IMPLIED
          present (normal|float|alert|altrep) #IMPLIED
          presspec CDATA #IMPLIED
          valign (top|middle|bottom) "top"
          TIM2 NMTOKEN #IMPLIED >
```

%TCIFkb; T.8

An entity reference to a set of TCIF standard character entity references for keyboard-key characters, most of which can be mapped to characters in the KeycapExtrasTCIF font. It also includes "newline" and "Service Mark".

```
<!ENTITY % TCIFkb PUBLIC "-//USA/TCIF//ENTITIES Keybd-2//EN">
%TCIFkb;
```

This is the entity set:

```
<!-- Keybd-2 - TCIF Entity-reference set for keycaps - standard.
     Release: 2.4, 1997-07-16 (tcif_kb.ent) -->
<!-- Copyright (c) 1995-1997, Alliance for Telecommunications Industry
     Solutions (ATIS). Permission is hereby granted to use, copy, modify
     and distribute this material for any purpose and without fee,
     provided this notice continues to appear in all copies.
     Reproduction and distribution for resale is prohibited. -->
<!ENTITY newline      SDATA "[newline]"      --=newline-->
<!ENTITY smark        SDATA "[smark ]"       --=service mark (SM)-->
<!ENTITY hyph         SDATA "[hyph]"         --=literal hyphen-->
<!ENTITY again-key    SDATA "[again-key]"    --=again key-->
<!ENTITY alt-key      SDATA "[alt-key]"      --=alt key-->
<!ENTITY altg-key     SDATA "[altg-key]"     --=alt graph key-->
<!ENTITY amp-key      SDATA "[amp-key]"      --=ampersand key-->
<!ENTITY apple-key    SDATA "[apple-key]"    --=apple key-->
<!ENTITY break-key    SDATA "[break-key]"    --=break key-->
<!ENTITY bs-key       SDATA "[bs-key]"       --=backspace key-->
<!ENTITY caps-key     SDATA "[caps-key]"     --=capslock key-->
<!ENTITY clear-key    SDATA "[clear-key]"    --=clear key-->
<!ENTITY clover-key   SDATA "[clear-key]"    --=clover (option) key-->
<!ENTITY cmd-key      SDATA "[cmd-key]"      --=command key-->
<!ENTITY color-key    SDATA "[color-key]"    --=solid-color blank key-->
<!ENTITY compose-key  SDATA "[compose-key]"  --=compose [char] key-->
<!ENTITY copy-key     SDATA "[copy-key]"     --=copy key-->
<!ENTITY cr-key       SDATA "[cr-key]"       --=carriage return key-->
<!ENTITY ctrl-key     SDATA "[ctrl-key]"     --=control key-->
<!ENTITY cut-key      SDATA "[cut-key]"      --=cut key-->
<!ENTITY del-key      SDATA "[del-key]"      --=delete key-->
<!ENTITY delbk-arrow  SDATA "[delbk-arrow]"  --=del-back-arrow key-->
<!ENTITY delfd-arrow  SDATA "[delfd-arrow]"  --=del-forward-arrow key-->
<!ENTITY dnlft-arrow  SDATA "[dnlft-arrow]"  --=down-left-arrow key-->
<!ENTITY dnrt-arrow   SDATA "[dnrt-arrow]"   --=down-right-arrow key-->
<!ENTITY do-key       SDATA "[do-key]"       --=do key-->
<!ENTITY down-arrow   SDATA "[down-arrow]"   --=down-arrow key-->
<!ENTITY eight-key    SDATA "[eight-key]"    --=8-* key-->
<!ENTITY end-key      SDATA "[end-key]"      --=end key-->
<!ENTITY enter-key    SDATA "[enter-key]"    --=enter key-->
<!ENTITY esc-key      SDATA "[esc-key]"      --=escape key-->
<!ENTITY f1-key       SDATA "[f1-key]"       --=function key 1-->
<!ENTITY f2-key       SDATA "[f2-key]"       --=function key 2-->
<!ENTITY f3-key       SDATA "[f3-key]"       --=function key 3-->
<!ENTITY f4-key       SDATA "[f4-key]"       --=function key 4-->
<!ENTITY f5-key       SDATA "[f5-key]"       --=function key 5-->
<!ENTITY f6-key       SDATA "[f6-key]"       --=function key 6-->
<!ENTITY f7-key       SDATA "[f7-key]"       --=function key 7-->
<!ENTITY f8-key       SDATA "[f8-key]"       --=function key 8-->
<!ENTITY f9-key       SDATA "[f9-key]"       --=function key 9-->
<!ENTITY f10-key      SDATA "[f10-key]"      --=function key 10-->
<!ENTITY f11-key      SDATA "[f11-key]"      --=function key 11-->
<!ENTITY f12-key      SDATA "[f12-key]"      --=function key 12-->
<!ENTITY f13-key      SDATA "[f13-key]"      --=function key 13-->
<!ENTITY f14-key      SDATA "[f14-key]"      --=function key 14-->
<!ENTITY f15-key      SDATA "[f15-key]"      --=function key 15-->
<!ENTITY f16-key      SDATA "[f16-key]"      --=function key 16-->
<!ENTITY f17-key      SDATA "[f17-key]"      --=function key 17-->
```

```
<!ENTITY f18-key      SDATA "[f18-key]"      --=function key 18-->
<!ENTITY f19-key      SDATA "[f19-key]"      --=function key 19-->
<!ENTITY f20-key      SDATA "[f20-key]"      --=function key 20-->
<!ENTITY f21-key      SDATA "[f21-key]"      --=function key 21-->
<!ENTITY f22-key      SDATA "[f22-key]"      --=function key 22-->
<!ENTITY f23-key      SDATA "[f23-key]"      --=function key 23-->
<!ENTITY f24-key      SDATA "[f24-key]"      --=function key 24-->
<!ENTITY find-key     SDATA "[find-key]"     --=find key-->
<!ENTITY five-key     SDATA "[five-key]"     --=5-% key-->
<!ENTITY four-key     SDATA "[four-key]"     --=4-$ key-->
<!ENTITY front-key    SDATA "[front-key]"    --=front key-->
<!ENTITY funct-key    SDATA "[funct-key;"    --=function key-->
<!ENTITY help-key     SDATA "[help-key]"     --=help key-->
<!ENTITY hold-key     SDATA "[hold-key]"     --=hold key-->
<!ENTITY home-key     SDATA "[home-key]"     --=home key-->
<!ENTITY ins-key      SDATA "[ins-key]"      --=insert key-->
<!ENTITY left-arrow   SDATA "[left-arrow]"   --=left-arrow key-->
<!ENTITY lf-key       SDATA "[lf-key]"       --=linefeed key-->
<!ENTITY lt-key       SDATA "[lt-key]"       --=less-than key-->
<!ENTITY macro-key    SDATA "[macro-key]"    --=macro key-->
<!ENTITY meta-key     SDATA "[meta-key]"     --=meta key-->
<!ENTITY next-key     SDATA "[next-key]"     --=next key-->
<!ENTITY nine-key     SDATA "[nine-key]"     --=9-( key-->
<!ENTITY numlk-key    SDATA "[numlk-key]"    --=numlock key-->
<!ENTITY onoff-key    SDATA "[onoff-key]"    --=on/off key-->
<!ENTITY one-key      SDATA "[one-key]"      --=1-! key-->
<!ENTITY open-key     SDATA "[open-key]"     --=open key-->
<!ENTITY opt-key      SDATA "[opt-key]"      --=option key-->
<!ENTITY paste-key    SDATA "[paste-key]"    --=paste key-->
<!ENTITY pause-key    SDATA "[pause-key]"    --=pause key-->
<!ENTITY pf1-key      SDATA "[pf1-key]"      --=function key 1-->
<!ENTITY pf2-key      SDATA "[pf2-key]"      --=function key 2-->
<!ENTITY pf3-key      SDATA "[pf3-key]"      --=function key 3-->
<!ENTITY pf4-key      SDATA "[pf4-key]"      --=function key 4-->
<!ENTITY pf5-key      SDATA "[pf5-key]"      --=function key 5-->
<!ENTITY pf6-key      SDATA "[pf6-key]"      --=function key 6-->
<!ENTITY pf7-key      SDATA "[pf7-key]"      --=function key 7-->
<!ENTITY pf8-key      SDATA "[pf8-key]"      --=function key 8-->
<!ENTITY pf9-key      SDATA "[pf9-key]"      --=function key 9-->
<!ENTITY pf10-key     SDATA "[pf10-key]"     --=function key 10-->
<!ENTITY pf11-key     SDATA "[pf11-key]"     --=function key 11-->
<!ENTITY pf12-key     SDATA "[pf12-key]"     --=function key 12-->
<!ENTITY pf13-key     SDATA "[pf13-key]"     --=function key 13-->
<!ENTITY pf14-key     SDATA "[pf14-key]"     --=function key 14-->
<!ENTITY pf15-key     SDATA "[pf15-key]"     --=function key 15-->
<!ENTITY pf16-key     SDATA "[pf16-key]"     --=function key 16-->
<!ENTITY pf17-key     SDATA "[pf17-key]"     --=function key 17-->
<!ENTITY pf18-key     SDATA "[pf18-key]"     --=function key 18-->
<!ENTITY pf19-key     SDATA "[pf19-key]"     --=function key 19-->
<!ENTITY pf20-key     SDATA "[pf20-key]"     --=function key 20-->
<!ENTITY pf21-key     SDATA "[pf21-key]"     --=function key 21-->
<!ENTITY pf22-key     SDATA "[pf22-key]"     --=function key 22-->
<!ENTITY pf23-key     SDATA "[pf23-key]"     --=function key 23-->
<!ENTITY pf24-key     SDATA "[pf24-key]"     --=function key 24-->
<!ENTITY pgdn-key     SDATA "[pgdn-key]"     --=page down key-->
```

```
<!ENTITY pgup-key     SDATA "[pgup-key]"     --=page up key-->
<!ENTITY plain-key    SDATA "[plain-key]"    --=plain blank key-->
<!ENTITY prev-key     SDATA "[prev-key]"     --=prev key-->
<!ENTITY print-key    SDATA "[print-key]"    --=print key-->
<!ENTITY props-key    SDATA "[props-key]"    --=props key-->
<!ENTITY prtscr-key   SDATA "[prtscr-key]"   --=print screen key-->
<!ENTITY ret-arrow    SDATA "[ret-arrow]"    --=return-arrow key-->
<!ENTITY right-arrow  SDATA "[right-arrow]"  --=right-arrow key-->
<!ENTITY scrlk-key    SDATA "[scrlk-key]"    --=scroll lock key-->
<!ENTITY sel-key      SDATA "[sel-key]"      --=select key-->
<!ENTITY setup-key    SDATA "[setup-key]"    --=setup key-->
<!ENTITY seven-key    SDATA "[seven-key]"    --=7-& key-->
<!ENTITY shift-key    SDATA "[shift-key]"    --=shift key-->
<!ENTITY six-key      SDATA "[six-key]"      --=6-^ key-->
<!ENTITY space-key    SDATA "[space-key]"    --=spacebar-->
<!ENTITY stop-key     SDATA "[stop-key]"     --=stop key-->
<!ENTITY sysrq-key    SDATA "[sysrq-key]"    --=sysrq key-->
<!ENTITY tab-key      SDATA "[tab-key]"      --=tab key-->
<!ENTITY tab-arrow    SDATA "[tab-arrow]"    --=tab-arrow key-->
<!ENTITY three-key    SDATA "[three-key]"    --=3-# key-->
<!ENTITY two-key      SDATA "[two-key]"      --=2-@ key-->
<!ENTITY undo-key     SDATA "[undo-key]"     --=undo key-->
<!ENTITY uplft-arrow  SDATA "[uplft-arrow]"  --=up-left-arrow key-->
<!ENTITY up-arrow     SDATA "[up-arrow]"     --=up-arrow key-->
<!ENTITY uprt-arrow   SDATA "[uprt-arrow]"   --=up-right-arrow key-->
<!ENTITY window-key   SDATA "[zero-key]"     --=Windows-symbol key-->
<!ENTITY zero-key     SDATA "[zero-key]"     --=0-) key-->
```

TelDoc (TIM 1 only)

Renamed **TDoc** (T.9).

<TDoc> T.9

Normally the top-level element of any document marked up according to this DTD, and never anything else. According to SGML conventions, when the *element* **TDoc** is used, the *term* "TDoc" also occurs in the DOCTYPE declaration and is considered the document type.

The **TDoc** element contains the entire document, which consists, by definition, of a **DocID** complex, optional **Resources** (R.14) and **FrontMatter** (F.10) subdivision, a required **Body** (B.4), and optional **AppMatter** (A.11) and **BackMatter** (B.1) subdivisions. Those in turn, can contain paras and structural groupings, contents lists, sections, and/or indexes.

The required subelement **DocID** contains the document number and other identifying information. In a TEDD package, the content of **DocID** is repeated in a separate file, always named `docid`. (The TEDD package is described in TCIF-IPI-95-001.)

```
<!ELEMENT TDoc - - (DocID,Resources?,FrontMatter?,Body,AppMatter?,
          BackMatter?)>
<!ATTLIST TDoc
```

```
%baseatts;
%identity;>
```

See the entries for the entities **%baseatts;** (B.2) and **%identity;** (I.3) for all this element's attributes. No standard values have been established for the **type** attribute.

<TermTitle> (TIM 1 only)

Now the first or second **Title** in a **ListHead** (L.11).

<Tertiary> (TIM 1 only)

Replaced by a nested **IndexEntry** (I.10).

"Test", type= — LI (L.8)

TEX (notation) T.10

A text notation containing **TeX** markup (most likely of equations).

```
<!NOTATION TEX       PUBLIC "+//ISBN 0-201-13448-9::Knuth//NOTATION
          The TeXbook//EN">
```

It may be within a TIM file:

```
<Equation format="TEX"> ....
```

Or it may be in a separate file with a `.tex` extension, in a `tex` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s):

```
<!ENTITY eqn001 SYSTEM "../tex/eqn001.tex" NDATA TEX>

...

<Equation><ExternalText entityref="eqn001"></Equation>
```

The TEX format may be included in TEDD packages experimentally. Preferred formats will be specified in future TCIF Guidelines.

TEXT (notation) T.11

A text notation containing plain text without markup.

```
<!NOTATION TEXT       SYSTEM "Plain text">
```

It may be within a TIM file:

```
<Listing format="TEXT"> ....
```

Or it may be in a separate file with a `.txt` extension, in a `txt` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s):

```
<!ENTITY file001 SYSTEM "../txt/file001.txt" NDATA TEXT>
```

```
...
<Listing><ExternalText entityref="file001"></Listing>
```

%textatts; T.12

An entity defining attributes available for elements that can contain character data (text content):

```
<!ENTITY % textatts
         "format NOTATION (SGML|HTML|RTF|TEX|TBL|EQN|PIC|EPS|PS|PBM|PGM
            |PPM|CGM-CLEAR|TEXT) #IMPLIED
          sortas CDATA #IMPLIED" >
```

Whenever "%textatts;" appears in an element's ATTLIST, the two attribute declarations above are substituted for it. Refer to the individual entries for the meanings and uses of each: **format** (F.7) and **sortas** (S.16).

<TFoot> T.13

An optional collection of rows intended to provided a footer (possibly a running footer that repeats page by page) for a **TGroup** (which holds the content of a table or a portion of a table, see next element). Column specifications will be inherited from the **TGroup** if not specified within the **TFoot**.

Note (in the **TGroup** declaration, T.14) that the **TFoot** must appear before the **TBody**, so that even an unsophisticated rendering application can output the footer whenever it is called for.

```
<!ELEMENT TFoot - - (ColSpec*,Pt*,Row,(Pt|Row)*) >
<!ATTLIST TFoot
          id ID #IMPLIED
          type NMTOKENS #IMPLIED
          dstype CDATA #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          group NMTOKENS #IMPLIED
          status CDATA #IMPLIED
          version IDREFS #IMPLIED
          revstat (norev|revised|revflag|insert|insflag|delete|strike)
            #IMPLIED
          lang CDATA #IMPLIED
          present (normal|float|alert|altrep) #IMPLIED
          presspec CDATA #IMPLIED
          valign (top|middle|bottom) "top"
          TIM2 NMTOKEN #IMPLIED >
```

<TGroup> T.14

Defines the column specifications (and default column spanning) for a table or a portion of a table (each time columns or headers or footers change, there must

be a new **TGroup**. Also contains a **TBody** element (T.7) and optional **THead** (T.16) and **TFoot** (T.13), which in turn contain rows, which contain cells.

```
<!ELEMENT TGroup - - (ColSpec*,SpanSpec*,THead?,TFoot?,TBody) >
<!ATTLIST TGroup
          id ID #IMPLIED
          type NMTOKENS #IMPLIED
          dstype CDATA #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          group NMTOKENS #IMPLIED
          status CDATA #IMPLIED
          version IDREFS #IMPLIED
          revstat (norev|revised|revflag|insert|insflag|delete|strike)
            #IMPLIED
          lang CDATA #IMPLIED
          present (normal|float|alert|altrep) #IMPLIED
          presspec CDATA #IMPLIED
          align (left|right|center|justify|char) "left"
          char CDATA ""
          charoff NMTOKEN "50"
          cols NMTOKEN #REQUIRED
          colsep NMTOKEN #IMPLIED
          rowsep NMTOKEN #IMPLIED
          tgroupstyle NMTOKEN #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

Use of the **type** attribute is not supported.

tgroupstyle= T.15

Table group style name. Applies to **TGroup** only. Specifies a style defined externally, presumably in a FOSI, from which a particular **TGroup** can inherit formatting defaults. The default value is undefined.

```
          tgroupstyle NMTOKEN #IMPLIED
```

<THead> T.16

An optional collection of rows providing a header (possibly a running header that repeats page by page) for a **TGroup** (which holds the content of a table or a portion of a table, see T.14). Column specifications will be inherited from the **TGroup** if not specified within the **THead**.

```
<!ELEMENT THead - - (ColSpec*,Pt*,Row,(Pt|Row)*) >
<!ATTLIST THead
          id ID #IMPLIED
          remap NMTOKEN #IMPLIED
          type NMTOKENS #IMPLIED
          dstype CDATA #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
```

```
group NMTOKENS #IMPLIED
status CDATA #IMPLIED
version IDREFS #IMPLIED
revstat (norev|revised|revflag|insert|insflag|delete|strike)
  #IMPLIED
lang CDATA #IMPLIED
present (normal|float|alert|altrep) #IMPLIED
presspec CDATA #IMPLIED
valign (top|middle|bottom) "bottom"
TIM2 NMTOKEN #IMPLIED >
```

TIFF (notation) T.17

Notation that should be declared for a `filename.tif` bitmap graphic (Aldus Tagged Image File Format).

```
<!NOTATION TIFF     PUBLIC "+//ISBN 0-7923-9432-1::Graphic
           Notation//NOTATION Aldus/Microsoft Tagged Interchange File
           Format//EN">
```

Example:

```
<!ENTITY fig001 SYSTEM "../tiff/fig001.tif" NDATA TIFF>
```

The file should be stored in a `tiff` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s). Currently the TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG (P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18).

TIFF files come in very many varieties. Most of them are not acceptable under TCIF Guidelines (documented in TCIF-IPI-96-004), either because they use poor compression methods (and so are unnecessarily large) or they use features that are not supported by most applications:

- uncompressed files — use LZW compression instead
- files with "RLE" (or "Packbits") compression — use LZW compression instead
- files with CCITT Group 3 compression — use LZW or CCITT Group 4 compression instead
- CMYK (Version 6.0) — use RGB or "Class R" TIFF instead
- YCbCr (Version 6.0) — use RGB or "Class R" TIFF instead
- JPEG (Version 6.0) — use JFIF format (J.1) instead
- Group 3, "FAX", or "Class F" — use "Class G" TIFF or, if necessary, TIFFGRP4 format (T.18)

CCITT Group 4 compression also presents problems for many applications, but there's no good substitute — it is the best compression for large black-and-white images, like scanned engineering drawings — so it's treated as a special case (T.18).

It's very difficult to determine what variety a TIFF file is, since the information is in a dozen or so binary-coded tags in no fixed position. It's best to try to choose wisely from the options your application gives you when creating the file. Usually these varieties are OK: "Version 5.0,""Class B" (black & white), "Class G" (grayscale), "Class P" (paletted), "Class M" (Macintosh), "Class R" (RGB or "True Color"). Compression must always be LZW for TCIF compliance. Any options for dithering or palette type are purely aesthetic.

TIFFGRP4 (notation) T.18

Notation that should be declared for a `filename.tif` graphic (Aldus Tagged Image File Format) with CCITT Group 4 compression.

```
<!NOTATION TIFFGRP4  SYSTEM "Black & white TIFF with Group IV
          compression">
```

Example:

```
<!ENTITY fig001 SYSTEM "../tiff/fig001.tif" NDATA TIFFGRP4>
```

The file should be stored in a `tiff` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s). Currently the TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG (P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18).

Group 4 TIFF should be used only for large, detailed black-and-white images, like scanned engineering drawings, requiring very efficient file compression. The trade-off for its excellent compression is that only a small percentage of graphics applications can display files in this format — if you set up a low-cost viewer as a helper application for TIFF files, odds are it won't open this kind. Therefore it has a separate notation from other TIFF files, allowing it to be assigned to a different application.

The TIFF format standard is so flexible that there are infinite varieties even of "Group 4" TIFF. You should check your variety with some of the viewers known to handle typical Group 4 files: in MS Windows, those include MS Word and PowerPoint, HiJaak PRO, QV, VuePrint, Graphic Workshop, AlterImage, and Graphic PRO. Even some of those may differ on the basic question of which color is black and which is white.

<TIM2> T.19

An attribute available for virtually every TIM element but having no use in TIM. This attribute would be needed by any DTD derived from TIM, to show what TIM element a given element in the derived DTD should be translated into. It is the complement of TIM's **`remap`** attribute. It is defined for TIM only so that it will not have to deleted in translating from the derived markup to TIM.

```
          TIM2 NMTOKEN #IMPLIED
```

This attribute is defined for all elements that have the **%baseatts;** (B.2).

"time", quantum= (Q.1)

<Title> T.20

A title of any element from a **TDoc** to a list item. It may appear without **TitleGroup** (T.21) around it, or as the only subelement of a **TitleGroup**, or accompanied by optional **SuperTitles**, **SubTitles**, and a **ShortTitle**; in a list item it can only appear alone. Titles do not include numbers, which are attributes of the element the title belongs to.

```
<!ELEMENT Title - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef
          |IndexTerm|Pt|Graphic|Object|ExternalText|SimpleList|Equation
          |GraphicalText)* >
<!ATTLIST Title
          %baseatts;
          %commonatts;
          %textatts; >
```

Generally, what needs to be known about the type of **Title** is clear from the context (what it is the title of), so there are no established values for **type**. See the entries for the entities **%baseatts;** (B.2), **%commonatts;** (C.19), and **&textatts;** (T.12) for all this element's attributes.

<TitleGroup> T.21

Except for list items and list groups, elements that can have a title can also have one or more supertitles or subtitles, and can also have a short version of the title (provided, say, for a running header). All of these elements are enclosed in a **TitleGroup**.

```
<!ELEMENT TitleGroup - - (SuperTitle*,Title+,SubTitle*,ShortTitle*) >
<!ATTLIST TitleGroup
          %baseatts; >
```

Use of the **type** attribute is not supported. See the entry for the entity **%baseatts;** (B.2) for all this element's attributes.

"Tool", type= — T (T.1)

"top", frame= (F.9)

"top", valign= (V.1)

"topbot", frame= (F.9)

"TradeAcknowl", type= — Section (S.4)

"TrdMk", type= — T (T.1)

<TreeLoc> T.22

HyTime element that creates links to "unnamed" elements (elements that do not have IDs). **TreeLoc** specifies these elements by pointing to their nesting level within the tree structure of the document. When creating a link to an unnamed element, a **Ref** is used to point to the **TreeLoc** element. The **TreeLoc**, in turn, tracks an element's nesting path in the tree structure through a series of counters.

```
<!ELEMENT TreeLoc - - (#PCDATA) >
<!ATTLIST TreeLoc
          id ID #REQUIRED
          HyTime NMTOKEN #FIXED "TreeLoc"
          locsrc IDREFS #IMPLIED >
```

The **locsrc** attribute is used to specify the ID of the top-level element in the tree (for internal links), or the ID of a **NameLoc** element (for external links).

Example for an internal cross-reference:

```
<Ref rid="TCIF-IPI-95-004-XREF-1">...</Ref><TreeLoc id="TCIF-
IPI-95-004-XREF-1" locsrc="TCIF-IPI-95-004-Section-3">1 4 1 1
2</TreeLoc>
```

This example could link to (or retrieve) a list item in Section 3, which is presumably the first ancestor element with an ID. It counts the first element at the hierarchical level of the element with the ID "TCIF-IPI-95-004-Section-3", which is that element itself, then its fourth child element (maybe a the third **P** after the section's **TitleGroup**), then that element's first child (maybe an **OrderedList**), then that element's first child (**OrdListGrp**), then that element's second child (second **LI**).

Example for an external cross-reference (the external document must be given an entity name in the current document's prologue):

```
<!DOCTYPE TDoc PUBLIC "-//USA/TCIF//DTD TIMM-1//EN" [

...

<!ENTITY Tedd SYSTEM "../../../ipi95001/mtext/main.tim" NDATA
SGML>
```

```
...
]>
<TDoc>
...
...<Ref rid="TCIF-IPI-95-004-XREF-2"> (see "The TEDD
Package")</Ref><TreeLoc id="TCIF-IPI-95-004-XREF-2"
locsrc="TCIF-IPI-95-004-NAMELOC-1">1 4 1 1 2 </
TreeLoc><NameLoc id="TCIF-IPI-95-004-NAMELOC-1"><NmList
docorsub="Tedd">TCIF-IPI-95-001-XREF-342</NmList></NameLoc>
```

type= T.23

Indicates the primary semantic type of an element. Examples:

- To indicate that a **Section** is a Glossary (for which there is no specific element): **<Section type="Glossary">**
- To specify that an **Annote** is a footnote (floating note) rather than a fixed note: **<Annote type="FNote">**
- To mark up a quotation:
  - in-line: **<T type="Quote">**
  - a paragraph: **<P type="Quote">**
  - more than a paragraph: **<S type="Quote"><P> ... </P><P> ....**

```
type NAMES #IMPLIED
```

Default value: undefined.

This attribute is defined for all elements that have the **%baseatts;** (B.2).

Some entries here for individual elements list the values of **type** that rendering applications are expected to recognize for that element. Others may be used, but recipients will need to be made aware of their presence and their meaning. Values are SGML NMTOKENs, so they must be one word (no space), but case is not distinguished ("DOCID" = "DocID" = "docid"). Although the attribute value can be more than one NMTOKEN, such values are not standard so far, and there should be no general expectation that they would be interpreted correctly.

types= T.24

Specifies the values of the **type** or **dstype** attributes that elements should have to be listed in a **Contents** list or an **Index**. See **Contents** (C.22) for more explanation.

```
types NMTOKENS #IMPLIED
```

The default value is undefined: values of **type** and **dstype** should be ignored and all elements that match the remaining specifications should be included.

U (UnorderedList — use)

"und", emph= (E.6)

"unified", nametype= (N.6)

<UnorderedList> U.1

An ordinary list in which items are marked by a character or string (normally the same for each item) rather than a sequence of numbers or letters — compare **OrderedList** (O.3) and **SimpleList** (S.15). Its items may be grouped using the **UnordListGrp** element (U.2), each of which can provide titles or specify a different mark for the list items. The numbering attributes of the **UnorderedList**, if specified, are applied to the list itself, as a block, not to the items. (The list would have to be one of a related group of lists, indicated by the **type** attribute, to be numbered meaningfully.)

```
<!ELEMENT UnorderedList - - ((Title|TitleGroup)?,ListHead?,
          (UnordListGrp+|(Danger|Caution|Warning|Admon|Annote|LI
          |Pt)*)) >
<!ATTLIST UnorderedList
          %baseatts;
          %identity;
          mark CDATA #IMPLIED
          %commonatts; >
```

No values have been established for the **type** attribute. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), and **%commonatts;** (C.19) for more this element's attributes.

<UnordListGrp> U.2

A section of an **UnorderedList** (U.1). It specifies the list mark for the items within it and may have a title that distinguishes it from the other sections of the list or different headings for the mark and content of the list items within it. **UnordListGrp** does not allow the list to be discontinuous; a discontinuous ("distributed") list must use **DistOrdLI** items (D.6). **UnordListGrp** is analogous to **TGroup** within tables. The mark on the items is specified by the **mark** attribute (M.1).

```
<!ELEMENT UnordListGrp - - (Title?,ListHead?,(Danger|Caution|Warning
          |Admon|Annote|LI|Pt)*) >
<!ATTLIST (UnorderedList|UnordListGrp)
          %baseatts;
          %identity;
          mark CDATA #IMPLIED
          %commonatts; >
```

Use of the **type** attribute is not supported. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), and **%commonatts;** (C.19) for more on this element's attributes.

url= U.3

Uniform Resource Locator (a World Wide Web address). Used by **URLRef** elements (U.4) to identify the objects they point to.

```
url CDATA #REQUIRED
```

Example:

```
... <URLRef url="ftp://www.bellcore.com/pub/TCIF/ipi_tedd/
ipi95001/mtext/main.tim"> (see TCIF-IPI-95-001)</URLRef>.
```

<URLRef> U.4

Simple reference to a Uniform Resource Locator on the World Wide Web. Its content could be empty, the URL, or a text description. The URL to be used for hyperlinking is the value of the **url** attribute, not the content of the **URLRef**. The **include** attribute can be used to specify what the textual content of the element should be if the supplied content is inappropriate (e.g., a page number). A reference to an element within the working document is a **Ref** (R.2).

```
<!ELEMENT URLRef - - (#PCDATA|T|Sub|Sup|Emph|Symbol|Ref|URLRef|IndexTerm
          |Pt|Graphic|Object|ExternalText|SimpleList|Equation
          |GraphicalText)* >
<!ATTLIST URLRef
          %baseatts;
          %identity;
          %commonatts;
          %textatts;
          url CDATA #REQUIRED
          target NMTOKEN #IMPLIED
          prefix CDATA #IMPLIED
          infix CDATA #IMPLIED
          suffix CDATA #IMPLIED
          include NMTOKENS #IMPLIED >
```

No values have been established for the **type** attribute. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%commonatts;** (C.19), and **&textatts;** (T.12) for all this element's attributes.

use= U.5

Tells whether to include or ignore content specific to a particular version of the document (identified by **Version** (V.7)).

```
use (include|ignore) #IMPLIED
```

"utc", quantum= (Q.1)

V (valign — versions)

"Value", type= — ListTerm (L.14) and T (T.1)

valign= V.1

Applies to **THead**, **TFoot**, **TBody**, **Row**, and **Entry**. Specifies whether cell contents should align to the top, center, or bottom when other cells in the row are taller. The definition for **THead** is:

```
valign (Top|Middle|Bottom) "Bottom"
```

The definition for **TBody** and **TFoot** is:

```
valign (Top|Middle|Bottom) "Top"
```

The definition for **Row** and **Entry** is inherited unless specified:

```
valign (Top|Middle|Bottom) #IMPLIED
```

<VariableList> V.2

A list in which items begin with varying terms ("headwords") — often a glossary, explanation of error messages, etc. The definition of **VariableList** permits a list label (number) as well as the headword on each item, and headings over the mark, the term, and the rest of the item.

Like most lists, **VariableList** can be divided into one or more sections, **VarListGrp**s, which define or redefine the list-item numbering, if any, and the subtitles over the parts of the list. The **VariableList**'s own numbering attributes apply only if there are no **VarListGrp**s.

```
<!ELEMENT VariableList - - ((Title|TitleGroup)?,ListHead?,(VarListGrp+
          |(Danger|Caution|Warning|Admon|Annote|VarLI|Pt)*)) >
<!ATTLIST (SegListGrp|OrderedList|OrdListGrp|VariableList|VarListGrp)
          %baseatts;
          %identity;
          %numbering;
          %commonatts; >
```

No values have been established for the **type** attribute. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), and **%commonatts;** (C.19) for more on this element's attributes.

For a glossary, use **Ref** cross-references in the text that point to the **VarLI**s:

```
... TIM is an implementation of <Ref target="VarLI" rid="TCIF-
IPI-95-004-GLOSS-1234">SGML</Ref>.

...

... <VarLI id="TCIF-IPI-95-004-GLOSS-1234"><ListTerm>SGML</
ListTerm><LI><P>Standard Generalized Markup Language ...</P></
LI></VarLI>
```

Comment

There are several ways a rendering application could choose to lay out a variable list (the following examples may not be formatted as intended in any but the printed version of this document):

Term Definition of the term. More definition of the term. Some more definition of the term.

Much lengthier term Definition of the term. More definition of the term. Some more definition of the term.

Or:

Term	Definition of the term. More definition of the term. Some more definition of the term.
Much lengthier term	Definition of the term. More definition of the term. Some more definition of the term.

Or

Term	Definition of the term. More definition of the term. Some more definition of the term.
Much lengthier term	Definition of the term. More definition of the term. Some more definition of the term.

Or

Term

Definition of the term. More definition of the term. Some more definition of the term.

Much lengthier term

Definition of the term. More definition of the term. Some more definition of the term.

Or

Word: **Term**

Meaning: Definition of the term. More definition of the term. Some more definition of the term.

Word: **Much lengthier term**

Meaning: Definition of the term. More definition of the term. Some more definition of the term.

In the last example, the labels would be supplied by the list's **`TermTitle`** and **`LITitle`**, which could also appear as column headings in any of the other renderings.

<VarLI> V.3

An item of a **VariableList** (V.2). Contains at least one **ListTerm** ("headword"), and at least one **LI** list item.

```
<!ELEMENT VarLI - - (ListTerm+,Pt*,LI,(Pt|LI)*) >
<!ATTLIST VarLI
          %baseatts;
          %identity;
          %commonatts; >
```

Use of the **type** attribute is not supported (values have been established for **ListTerm**, L.14). See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), and **%commonatts;** (C.19) for all this element's attributes.

Comment

Note the "+" signs in the element declaration. Why can there be more than one **ListTerm** or **LI**? It seems convenient to allow one definition to apply to two synonymous terms, or two different definitions to be applied to one term. (The DocBook definition of **VarLI** allows multiple terms but not multiple definitions.) Of course, this complicates the decisions about presentation that need to be made by the rendering application.

<VarListGrp> V.4

A section of a **VariableList** (V.2). It may have a title that distinguishes it from the other sections of the list or different headings for the term and content of the list items within it, or it may specify numbering for the items within it. **VarListGrp** does not allow the list to be discontinuous; a discontinuous ("distributed") list must use **DistVarLI** items (D.7). The numbering of the items, if any, is specified by the numbering attributes in **VarListGrp**. The group itself, though it can have a title, cannot have a mark or number.

```
<!ELEMENT VarListGrp - - (Title?,ListHead?,(Danger|Caution|Warning|Admon
          |Annote|VarLI|Pt)*) >
<!ATTLIST VarListGrp
          %baseatts;
          %identity;
          %numbering;
          %commonatts; >
```

Use of the **type** attribute is not supported. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), and **%commonatts;** (C.19) for all this element's attributes.

verbatim (notation, TIM 1 only) — Listing (L.12)

<VersDate> V.5

A required subelement of **Version** (V.7) giving the date of the particular version identified by it. The preferred (but not required) format for all dates in TIM is YYYY-MM-DD (with day optional).

```
<!ELEMENT VersDate - - (#PCDATA) >
<!ATTLIST VersDate
          id ID #REQUIRED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

<VersDesc> V.6

May occur any number of times in **Version** (V.7) to describe the particular version identified by it. The form and content are left to the discretion of the document originator.

```
<!ELEMENT VersDesc - - (#PCDATA) >
<!ATTLIST VersDesc
          id ID #REQUIRED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

<Version> V.7

Identifies a particular version of a document. TIM allows a single document instance to include more than one version of the document. **Version** elements identify the included versions, and their **use** attributes (U.5) tell which one(s) to use in the current document instance. Any element of the document that is specific to one version should have its **version** (V.8) attribute point to the **Version** element identifying that version.

```
<!ELEMENT Version - - (Au*,VersDate,VersDesc*) >
<!ATTLIST Version
          id ID #REQUIRED
          use (include|ignore) #IMPLIED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

version= V.8

Points to the **Version** element (V.7) that identifies the particular version of the document the current element belongs to. The referenced **Version** element's **use** attribute (U.5) says whether the current element should be included or ignored in the current document instance.

```
version IDREFS #IMPLIED
```

This attribute is defined for all elements that have the **%baseatts;** (B.2).

<VersionControl> V.9

A holder for any and all **Version** elements (V.7).

```
<!ELEMENT VersionControl - - (Version)+ >
<!ATTLIST VersionControl
          id ID #REQUIRED
          remap NMTOKEN #IMPLIED
          remapatt CDATA #IMPLIED
          remapto CDATA #IMPLIED
          TIM2 NMTOKEN #IMPLIED >
```

versions= V.10

Specifies what values of the **version** attribute an element should have to be listed in a **Contents** list or an **Index**. See **Contents** (C.22) for more explanation.

```
versions CDATA #IMPLIED
```

The default value of this attribute is undefined — that is, all values of **version** should be included.

"vert", arrange= (A.12)

See **arrange** (A.12).

"Volume", type= — T (T.1)

W (Warning — WPG)

<Warning> W.1

An admonishment that normally warns of the risk of equipment damage, software corruption, or data loss. Admonishments — the other specific types are **Admon** (A.2), **Caution** (C.3) and **Danger** (D.1) are important in telecom documents, and ideally they should be given sophisticated presentation: they generally apply to a span of text, and should appear automatically whenever any part of that text is presented (see the comment at **Caution**).

```
<!ELEMENT Warning - - (Title?,(P|S|Annote|Frame|Table|OrderedList
          |UnorderedList|VariableList|SegmentedList|Listing|Pt|IndexTerm
          |Graphic|Object|Flowchart|ExternalText)*) >
<!ATTLIST Warning
          %baseatts;
          %identity;
          %numbering;
          %commonatts;>
```

Use of the **type** attribute is not supported. See the entries for the entities **%baseatts;** (B.2), **%identity;** (I.3), **%numbering;** (N.13), and **%commonatts;** (C.19) for all this element's attributes. It is expected that the meanings, uses, and presentation specifications of admonishments will be the subject of a separate TCIF Guideline.

WAV (notation) W.2

Notation that should be declared for a `filename.wav` audio file.

```
<!NOTATION WAV       SYSTEM "WAV">
```

Example:

```
<!ENTITY clip001 SYSTEM "../wav/clip001.wav" NDATA WAV>

...

<Object entityref="clip001">
```

The file should be stored in a `wav` directory at the same level as the `mtext` directory containing the TIM file(s). WAV files may be included in TEDD packages experimentally. Preferred formats will be specified in future TCIF guidelines. Currently the options for sound files are AU (A.16), MIDI (M.5), and WAV.

width= W.3

Specifies the intended maximum width in characters of lines in a **Listing**.

```
          width NMTOKEN #IMPLIED
```

The default should be 80. A higher number warns the rendering application to use a smaller point size or a mechanism (such as a hanging indent) for wrapped lines.

WK1 (notation) W.4

Notation that should be declared for a `filename.wk1` spreadsheet file.

```
<!NOTATION WK1       SYSTEM "WK1 spreadsheet">
```

Example:

```
<!ENTITY sheet001 SYSTEM "../other/clip001.wk1" NDATA WK1>

...

<Object entityref="sheet001">
```

The file should be stored in an `other` directory at the same level as the `mtext` directory containing the TIM file(s). WK1 files, the lowest-common-denominator format for spreadsheets, may be included in TEDD packages experimentally. Preferred formats will be specified in a future TCIF Guideline.

WMF (notation) W.5

Notation that should be declared for a `filename.wmf` Windows MetaFile (metafiles may contain drawings, bitmaps, and text).

```
<!NOTATION WMF       PUBLIC "+//ISBN 0-7923-9432-1::Graphic
          Notation//NOTATION Microsoft Windows Metafile//EN">
```

Example:

```
<!ENTITY fig001 SYSTEM "../wmf/fig001.wmf" NDATA WMF>
```

The file should be stored in a `wmf` directory/folder at the same level as the `mtext` directory/folder containing the TIM document. WMF files may be included in TEDD packages experimentally. Preferred formats will be specified in future TCIF Guidelines. Currently the options are CGM-BINARY (C.6), CGM-CLEAR (C.7), DXF (D.35), EPS (E.12), PDF (P.5), PICT (P.9), WMF (W.5), and WPG (W.6).

"word", quantum= (Q.1)

WPG (notation) W.6

Notation that should be declared for a `filename.wpg` WordPerfect Graphic metafile (metafiles may contain drawings, bitmaps, and text).

```
<!NOTATION WPG       SYSTEM "WPG">
```

Example:

```
<!ENTITY fig001 SYSTEM "../wpg/fig001.wpg" NDATA WPG>
```

The file should be stored in a `wpg` directory/folder at the same level as the `mtext` directory/folder containing the TIM document. WPG files may be included in TEDD packages experimentally. Preferred formats will be specified in future TCIF Guidelines. Currently the options are BINARY (C.6), CGM-CLEAR (C.7), DXF (D.35), EPS (E.12), PDF (P.5), PICT (P.9), WMF (W.5), and WPG (W.6).

X (XLS — XWD)

XLS (notation) X.1

Notation that should be declared for a `filename.xls` spreadsheet file.

```
<!NOTATION XLS       SYSTEM "XLS spreadsheet">
```

Example:

```
<!ENTITY sheet001 SYSTEM "../other/clip001.xls" NDATA XLS>
...
<Object entityref="sheet001">
```

The file should be stored in an `other` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM file(s). XLS files may be included in TEDD packages experimentally. Preferred formats will be specified in future TCIF Guidelines.

?XML (processing instruction) X.2

Identifies the preferred character encoding as required for compatibility with XML (eXtensible Markup Language).

```
<?XML ECODING='UTF-8'>
```

XWD (notation) X.3

Notation that should be declared for a `filename.xwd` (X/Windows Dump) bitmap graphic file.

```
<!NOTATION XWD       PUBLIC "+//ISBN 0-7923-9432-1::Graphic
           Notation//NOTATION MIT X Consortium Window Dump//EN">
```

Example:

```
<!ENTITY fig001 SYSTEM "../xwd/fig001.xwd" NDATA XWD>
```

The file should be stored in an `xwd` directory/folder at the same level — having the same parent — as the `mtext` directory/folder containing the TIM document. XWD files may be included in TEDD packages that follow TCIF guidelines. Currently the TCIF-approved options for bitmap (raster) graphic files are GIF (4-bit, or 16-color, and 8-bit, or 256-color, GIF87a only, G.1), JFIF (JPEG, J.1), PNG (P.11), TIFF (Version 5.0 Classes B, G, P, and R, with LZW compression only, T.17), and TIFFGRP4 (T.18).