Guide to the DocBook DTD
Documentation version 1.0 for release 2.2.1

HaL Computer Systems, Inc. O'Reilly & Associates, Inc.

February 24, 1995

Contents

Acknowledgements

The DocBook DTD is a joint effort of Hal Computer Systems Inc., and O'Reilly & Associates, Inc. Conleth O'Connell, Eve Maler, and Terry Allen are the current design team for the DTD; Con deserves special mention as chief architect. Through several revisions, a lot of other people at several companies contributed their insight and encouragement, among them David Bass, Bryan Caporlette, Dale Dougherty, and Steve Williams.

0.1 How to Get the DocBook DTD Online

You can find the DocBook DTD and its documentation online in the Davenport archive (`/pub/davenport/docbook`) at `ftp.ora.com` (198.112.208.13).

This sample session shows how to retrieve the DTD and its documentation:

```
%ftp ftp.ora.com
Connected to amber.ora.com.
220 amber FTP server (Version wu-2.4(1) Fri Apr 15 14:14:30 EDT 1994) ready.
Name (ftp.ora.com:terry): anonymous
331 Guest login ok, send your complete e-mail address as password.
Password: ← type e-mail address
ftp>cd pub/davenport/docbook
```

The DocBook DTD and related ASCII files are in a file named `docbook.N.shar`, where N is the current revision number:

```
ftp>get docbook.2.2.1.shar
```

Most of these files also exist separately and may be ftp'd individually.

The *get* command will put this ASCII shar file on your system. You must later unpack it on your system:

```
sh docbook.2.2.1.shar
```

Back to your ftp session. The DocBook documentation is in a compressed PostScript file:

```
ftp>binary
ftp>get docbook.ps.Z
```

This puts the binary file `docbook.ps.Z` on your system. You must uncompress it before sending it to a PostScript printer or viewer.

Chapter 1

Development of the DocBook DTD

The DocBook DTD defines structural SGML markup for computer documentation and technical books. It is supported by the Davenport Group, which is a group of software documentation producers that was established to promote the interchange and delivery of computer documentation using SGML and other relevant standards.

Our primary goal in developing this DTD was to filter existing software documentation into SGML. It describes the structures we and other producers and consumers of software documentation have encountered in processing large bodies of such material. While it was initiated as an interchange format, the DTD is now coming into use as a production DTD.

We expect that this DTD will evolve as it is applied to more documents, as more SGML tools come into use, and as users gain more experience in working with it. We are interested in your use of the DocBook DTD and welcome comment (see title page for addresses). We are committed to reviewing revisions made by various groups and evaluating them for inclusion in the next release.

Groups using the DocBook DTD for material other than software documentation may need to extend it for their domain of applications. Likewise, anyone may wish to construct a subset of this DTD, or a local version that constrains authors more tightly. Before making changes you should read the section near the top of the DTD itself regarding local modifications.

1.1 Changes Since the Last Revision

Both the structure of the distribution and the content of the DTD have changed somewhat since the last public release, 2.1.

1.1.1 New Policy on Future Changes

We have established a new policy that we shall make no future backward incompatible changes in minor revisions (x.a to x.b), that backward incompatible changes shall be made only in major revisions (x.a to y.a), that there shall be no more than two major revisions per year, and that we shall give warning of these at least six months in advance.

Accordingly, some future backward incompatible changes are signalled in "FUTURE USE" comments in the DTD. We expect to make these changes in version 3.0:

- The SpanEnd attribute on IndexTerm will be renamed to indicate that it points to the content of the IndexTerm being spanned. Another attribute will be added to be used if it is desired to show that the IndexTerm is the origin point of a span.
- FootnoteRef will be made EMPTY and its Mark attribute renamed as Label.
- RevHistory will require at least one Revision.

1.1.2 Structure of the Distribution

The DocBook distribution is now composed of the DTD, an SGML declaration, a catalogue of public entities in the style defined in SGML Open Technical Report 9401, and this documentation.

The list of public entities formerly found in the DTD itself is now to be found in that catalogue, which may be used with parsers that can accept it (such as James Clark's SP).

Some small changes in the order of declarations in the DTD have been made.

This documentation is now split up into multiple files: one main file containing the narrative text, and one file for each of the SGML examples. The examples can now be written in SGML without escaping the control characters, and they can be validated by proper parsing. If you wish to print this document from the SGML source, be aware that these example files, referred to by use of the ULink element, need to be dealt with appropriately for your application.

1.1.3 Parameter Entities in the DTD

There has been some rearrangement of parameter entities. The BeginPage element and the parameter entity %ndxterm.gp; have been removed to a new parameter entity, %ubiq.gp;, which is provided as an inclusion in some content models, and excluded from others. This means that BeginPage and IndexTerm may appear anywhere within the contents of the elements in which %ubiq.gp; is included; these are the elements from the highest level of the DocBook hierarchy (Set) down to Sect1 and RefEntry. If you use BeginPage or IndexTerm, and wish to declare a doctype from a lower level of the hierarchy (such as Para), you may need to modify the DTD or redeclare the doctype element in the prologue to your instance, including %ubiq.gp;.

1.1.4 Backward Incompatible Changes

In this revision most changes are minor, although a few are backward incompatible. As remarked above, future backward incompatible changes are signalled in the DTD.

- The elements Character, Charset, Font, and Glyph have been removed, as have EventStructure, EventType, Mask, and ProtocolRequest.
- The FAXTILE notation has been removed.
- IndexTerms are no longer allowed in Highlights.
- Footnote is now excluded from itself (beware, all you folks who were footnoting your footnotes!)
- The Mark attribute on FootnoteRef (which is to be renamed Label in a future revision) is now IMPLIED rather than CONREF.
- The Width attribute has been removed from ScreenShot.
- Logo has been removed from the Class attribute on ProductName and Trademark.
- The Linkend attribute has been removed from ToCFront.

- The Linkend and PageNum attributes formerly found on the components of ToC have been moved to the new ToCEntry element.

1.1.5 Other Changes

Other significant nonincompatible changes in this revision include:

- A notation declaration has been added for CGM-BINARY.
- ProductName and ProductNumber are now available within %docinfo.content.gp;, and thus as in-line markup in most places.
- Optional is now available as in-line markup.
- Subtitle may now contain in-line markup.
- All the link elements may now contain in-line objects, and have common attributes. XRef now has common attributes in addition to Linkend and Endterm.
- LegalNotice may now contain many additional block-oriented elements.
- In Revision, AuthorInitials and RevRemark are now optional.
- Application, CiteTitle, and Quote may now contain a wider range of in-line elements.
- Subscript and Superscript may now contain Emphasis.
- GlossSee and GlossSeeAlso may now contain in-line elements.
- PrimaryIE, SecondaryIE, TertiaryIE, SeeIE, and SeeAlsoIE may now contain in-line elements, and their Linkends attributes are now IMPLIED rather than REQUIRED.
- Contrib has been created and included in the content models of Author, Editor, and OtherCredit, as a place to store information about the contributions of these parties to the document in question. It contains plain text.
- GlossList has been created: it's a set of GlossEntries and may be used anywhere lists are allowed.
- InformalExample has been created, as a parallel to other Informal block-oriented elements. It's just like Example, but without a title.
- ToCentry has been created and added to all the components of ToC, to hold the actual entries in a Table of Contents. It may contain plain text and in-line elements.

- Address now contains the new OtherAddr for modes of addressing not now covered by other elements.
- Block-oriented elements may now occur between SubSteps.
- Para may now properly contain InlineGraphic and InlineEquation.
- Entry (a table cell) may now contain elements from %code.example.gp; (LiteralLayout, ProgramListing, Screen, and ScreenShot).
- A Glossary may now close with an optional Bibliography.
- A RefNameDiv may now contain more than one RefClass.
- Editor and OtherCredit have been added as allowable contents in AuthorGroup.
- A new Scope attribute has been added to IndexTerm with the allowed values All, Global, and Local (the default is IMPLIED), to meet the case of IndexTerms that should be used to generate, *e.g.*, an index for a Book (Local) but not an index for a Set (Global) or vice versa; the value All should be used when the IndexTerm should appear in all generated indices. You should define your implementation's default accordingly.
- Also, a new Zone attribute, with IMPLIED IDREFS values, has been added to IndexTerm. This attribute is to be used when it is desired to attach indexing information to SGML elements rather than, or in addition to, placing IndexTerms in the flow of the text so as to be sure page references are generated correctly. Zone could be used to attach keywords to sections and paragraphs, for example, facilitating the assembly of nonstructural views of the indexed text.
- The RenderAs attribute default for Sect1–5 and Bridgehead has been changed to IMPLIED (from the same value as the element's generic identifier, or Other, in the case of Bridgehead–there might be some backward incompatibility for Bridgehead).
- A SrcCredit attribute has been added to Graphic, InlineGraphic, and LoTEntry for storage of information crediting the source of an image.
- Many more in-line elements now have common attributes.
- The ID attributes on Footnote and BeginPage are now IMPLIED rather than REQUIRED.
- The PageNum attribute on Anchor is now IMPLIED rather than REQUIRED.
- Part has been added as a defined value for the Pubwork attribute on CiteTitle.

- GlossEntry now has a SortAs attribute, like IndexEntry.
- The Subject attribute on GlossDef has been changed from NMTOKENS to CDATA.

1.2 Using the DocBook DTD

In order to parse your document, you must specify its highest-level element at its head, in the document type declaration:

```
<!DOCTYPE chapter SYSTEM "-//HaL and O'Reilly//DTD DocBook//EN">
```

or construct a shell file that contains a document type declaration, and refers to your document as an entity. In this case, your document should have no document type declaration itself.

Some SGML tools compilation a preparsed version of a DTD for internal use, and may fix the doctype at that time. The Docbook DTD as distributed may require some assembly for input to such a tool. For example, to prepare the DTD for compilation by ArborText's Docarch product for the doctype Book, prepend the SGML declaration to the DTD wrapped in a document type declaration:

```
<!SGML  "ISO 8879:1986"
   .
   .
   .
APPINFO
                  NONE
>
<!DOCTYPE book [
  (DTD as distributed goes here)
]>
```

Chapter 2

A Structural View of the DTD

The DTD uses the model of books to represent the content of documentation. The primary metainformation about a collection of documents is located at the Book level. A collection of Books can be a Set, but no larger structures are provided. A Book is composed of book components, such as Prefaces and Chapters and Appendices, some of which may be organized in Parts. Within these book components additional levels of sectioning, marked as Sect1 through Sect5, may occur.

A section is further organized by block-oriented elements such as Paras (paragraphs). Other block-oriented elements, such as lists, are more complex, requiring other elements within them. For example, a Listitem (an item in a list) may contain more than one paragraph, and lists may be nested within Listitems.

In addition to its hierarchical organization, the information may be marked up with elements that indicate nonlinear relationships, including IndexTerms, XRefs (cross references), and links.

This DTD also provides many general and specific in-line elements, some of which are peculiar to computer documentation.

The following sections show examples of how to combine the elements defined in the DocBook DTD. Following these examples is a reference section with brief definitions of all the elements.

- SGML element names are not case-sensitive. They are capitalized here for readability.

- A few elements appear in various contexts; for example, Title is used for the contents of headings at all levels.
- In some elements the dreaded SGML mixed content is allowed, but it should almost always be benign.
- Some attributes are not mentioned in the following description. For full details on attributes, see the alphabetical list of elements.

Chapter 3

Sets, Books, and Their Parts

Here's a diagram of a fairly vanilla Book:

```
Book
     BookInfo
     ToC
     LoT
     Preface
     Chapter
     Chapter
     Chapter
     Chapter
     Reference
     Appendix
     Appendix
     Glossary
     Bibliography
     Index
```

Books may have Chapters and Appendices grouped in Parts:

```
Book
     BookInfo
     Preface
```

```
   Part
 PartIntro
 Chapter
 Chapter
   Part
 PartIntro
 Chapter
 Chapter
 Chapter
   Part
 PartIntro
 Chapter
 Chapter
```

A reference manual might be arranged like this:

```
Book
     BookInfo
     ToC
     LoT
     Preface
     Reference
     Reference
     Reference
     Reference
     Appendix
     Appendix
     Appendix
     Appendix
     Index
```

A Set of Books may have a SetInfo and must contain two or more Books:

```
Set
     SetInfo
     Book
     Book
     Book
```

In this example a short Book is represented, with its later parts commented out:

```
<!DOCTYPE book SYSTEM "docbook.dtd"
```

```
[
!ENTITY   bookinfo SYSTEM "bookinfo.sgm"
!ENTITY   pref SYSTEM "pref.sgm"
!ENTITY   c1 SYSTEM "ch01.sgm"
!ENTITY   c2 SYSTEM "ch07.sgm"
!ENTITY   appa SYSTEM "appa.sgm"
!ENTITY   appb SYSTEM "appb.sgm"
!ENTITY   glossary SYSTEM "glossary.sgm"
]
<BOOK>
&bookinfo;
&pref;
&c1;
<!-- &amp;c2; -->
<!-- &amp;appa; -->
<!-- &amp;appb; -->
<!-- &amp;glossary; -->
</BOOK>
```

A Book may begin with a Title and TitleAbbrev (abbreviated title), followed in order by an optional BookInfo, an optional ToC (Table of Contents), any number of LoTs (List of Titles), any number of Prefaces, main contents, and back matter. The main contents are required, and must be one or more Parts; one or more Chapters followed by any number of References; one or more Articles; or one or more References. All back matter is optional, but must appear in order: any number of Appendices, a Glossary, a Bibliography, and any number of Indexes, followed by any number of LoTs, followed by an optional ToC.

Book has FPI and Label attributes; you can supply an SGML Formal Public Indentifier (FPI) as the value of the former. You might use the number of a Book, when it is part of a Set, in the latter.

The Label attribute, which is provided for other elements, too, allows you to "take a snapshot" of a document collection, or to use one particular state of it for reference. The Mark attribute on OrderedList and the BeginPage element do the same. You might want to use these elements to ensure that an on-line presentation of your document collection can be matched to a print version by standard reference methods (citing chapter number, number of an item in a list, page number). If you desire to autonumber your text you can simply disregard these attributes.

Part contains an optional DocInfo, a required Title, an optional TitleAbbrev, an optional PartIntro, followed by one or more book components.

You can use book components individually. For example, here is a Chapter standing on its own:

```
Chapter
     Title
     TitleAbbrev
     Para
     Para
```

```
<CHAPTER ID=CH-061-ch05-1>
<TITLE>An Introduction to the X Window System
</TITLE>
<TITLEABBREV>Intro to X
</TITLEABBREV>
<PARA>And so on.
</PARA>
</CHAPTER>
```

3.1 Preface, Chapter, Appendix, Bibliography, Glossary, and Index

A Chapter or Appendix may have a DocInfo, followed by a required Title and some text—at least a Sect, a paragraph, a list, RefEntry, or other such structure. It may also have a TitleAbbrev. The DocInfo may contain information relevant to that book part alone; information general to the collection(s) of which it is part should be placed in the collection(s)'s BookInfo(s).

A Preface, Glossary, or Bibliography must have a Title. Thus you can use the Preface element for a foreword or acknowledgements section and call a Glossary or Bibliography something else, such as "Key Words," or "Sources."

Bibliography, Glossary, and Index are also allowed to have more specialized contents: BiblioEntry for Bibliography, GlossEntry for Glossary, and IndexEntry for Index. These specialized book parts may be divided into BiblioDivs, GlossDivs, or IndexDivs, respectively. (An IndexEntry is part of an Index, whereas an IndexTerm is part of the text of a document, marking the term to be indexed.) Additionally, an IndexDiv may have a SegmentedList, to accommodate permuted indices. A Glossary may end with a Bibliography.

IndexEntries and IndexTerms are discussed later.

BiblioEntries may begin and end with BiblioMisc, between which there must be an ArtHeader, BookBiblio, or SeriesInfo. This makes it possible to use the same content model for metainformation and bibliographic reference. (In the future there should perhaps be some way to link from a BiblioEntry to a BookBiblio somewhere on one's local system.)

GlossEntries are pairs composed of a GlossTerm (followed optionally by an Acronym and an Abbrev), which may contain in-line elements, and either a GlossSee (a cross reference to another GlossEntry where the GlossTerm is actually defined) or a GlossDef, which may contain elements typical of a Listitem, followed optionally by a GlossSeeAlso (a cross reference to another GlossEntry where there is related information about the GlossTerm).

ToC, LoT, Bibliography, Glossary, and Index may be book components on their own or may occur at the beginning or end of the body of a Preface, Chapter, Appendix, or Sect. A RefEntry (the element for a man page) cannot be a book component, but can appear in the same places within Preface, Chapter, Appendix, or Sect. You could construct a Chapter this way:

```
Chapter
     Title
     Para
     Sect1
  Title
  Para
  Para
  Para
  Glossary
  Bibliography
     Sect1
  Title
  RefEntry
  RefEntry
  RefEntry
  RefEntry
  Index
```

3.2 Table of Contents

A ToC, or table of contents, may be a book component on its own or may occur within other book components. ToC is subdivided to follow the divisions of a book: following an optional DocInfo, Title, and TitleAbbrev, a ToC may have any number of ToCFronts, which are the entries for the front matter. Following the ToCFronts, if any, a ToC must have either one or more ToCParts (entries for Parts) or ToCChaps (entries for Chapters and Appendices), and may have any number of ToCBacks (entries for back matter). A ToCPart begins with one or more ToCEntries (a wrapper for the actual table of contents entry), then contains any number of ToCChaps. ToCEntry has a PageNum attribute, which may have the value of a physical page number. A ToCChap begins with one or more ToCEntries, then may have any number of ToCLevel1s, which are entries for Sect1s. A ToCLevel1 begins with one or more ToCEntries, then may have any number of ToCLevel2s, and so on down to ToCLevel5, which may have only one or more ToCEntries. Thus if you have a Table of Contents that shows section headings, the second-level entries are nested within the first-level entries, and so on. You could make a link of all or part of the contents of a ToCEntry.

```
<TOC>
<TITLE>Table of Contents</TITLE>
<TOCCHAP pagenum="iv">
<TOCENTRY>Acknowledgements</TOCENTRY>
<TOCLEVEL1 pagenum="iv">
<TOCENTRY>How to Get the DocBook DTD Online
</TOCENTRY>
</TOCLEVEL1>
</TOCCHAP>
<TOCCHAP pagenum="1">
<TOCENTRY>Development of the DocBook DTD
</TOCENTRY></TOCCHAP>
</TOC>
```

3.3 Lists of Titles (LoT)

An LoT is like a ToC except that it is used for lists of tables, figures, and so on, and has no hierarchy. An LoT contains LoTEntries, which could contain links, just like ToCEntries. LoTEntries have a PageNum attribute, and a SrcCredit attribute, the latter for providing a credit for the source of, e.g., an illustration.

3.4 BookInfo and DocInfo

A BookInfo contains metainformation about the collection of documents of which it is a part. BookInfo must contain a BookBiblio (a wrapper for any information that might appear in a bibliographic reference), followed by any number of LegalNotices, followed by any number of ModeSpecs (which are link processing information sets, pointed to by the LinkMode attribute of OLink, and collected here for common reference). BookInfo has a Contents attribute, the values of which are the IDs of the ToC, LoTs, Prefaces, Parts, Chapters, Appendixes, References, Glossary, Bibliography, and Indexes comprising the Book, in the order of their appearance.

BookBiblio contains all the information about a book that may be relevant to a bibliographical citation, so that it may also be used or referred to by BiblioEntry. In fact, all the information typically found on a title page's recto and verso are included in BookBiblio, except the LegalNotices, although only Title and AuthorGroup are required. BookBiblio may contain, in order: the required Title (the title of the Book), optional TitleAbbrev, Subtitle, and Edition, followed by one or more required AuthorGroups; then, optionally, either an ISBN followed by an optional VolumeNum (the number of the volume in a Set) or (in case you are using Book for a journal composed of Articles) an ISSN followed by optional VolumeNum, optional IssueNum, and an optional PageNums (the numbers of the pages contained in a given issue or volume). After these elements there may occur, again optionally and in order, one occurrence each of the labelling information InvPartNumber, ProductNumber, ProductName, PubsNumber, and ReleaseInfo. Then there may be any number of Pubdates followed by any number of Publishers, followed by an optional Copyright, then an optional SeriesInfo (information about a series the Book is published as part of). Following that there may be any number of Abstracts, any number of ConfGroups (wrappers for information about a conference from which the Book results), any number of ContractNums mixed with any number of ContractSponsors, and an optional PrintHistory followed by an optional RevHistory.

Some of these BookBiblio elements have subelements; see the alphabetical list of elements for details.

An individual book component may have a DocInfo, in which information pertinent only to that component should appear. DocInfo contains, in order: a required Title, optional TitleAbbrev and Subtitle, followed by one or more AuthorGroups, any number of Abstracts, an optional RevHistory, and any number of LegalNotices.

3.5 References

A Reference is a collection of RefEntries. Before its RefEntries, it has an optional DocInfo, a required Title, an optional TitleAbbrev, and an optional PartIntro. It may be a book component or may occur within a Part. For example:

```
Reference
     Title
```

```
PartIntro
RefEntry
RefEntry
RefEntry
RefEntry
RefEntry
```

3.6 Sections (Heads)

A Preface, Chapter, or Appendix may contain up to five levels of hierarchical sections called Sect1, Sect2, and so on. In a RefEntry, these are RefSect1 through RefSect3 (no further subheads). Headings must be nested properly: a Sect3, for example, is not allowed directly under a Sect1 without a Sect2 in between. Sects1–4 that contain lower-level Sects may be empty of other content. However, a Bridgehead (a heading not contained in a wrapper associating it with the text following, and independent of the Sect hierarchy) may be inserted between block-oriented elements anywhere, with its Renderas attribute set to an appropriate value. Sects also have a Renderas attribute, so that you may accomodate cases for which it is desired that, for example, all the headings in a book part appear as third-level heads:

```
<SECT1 Renderas="Sect3">
<TITLE>My Masquerade as a Third-Level Head
</TITLE>
```

The Docbook DTD does not allow you to put paragraphs, lists, and other block-oriented elements between Sects.

A section includes a Title (the text of the heading itself) and all material following, up to the end of the document or the next section at the same level or the end of the higher-level section that contains the section in question. Consequently, anything may occur in a section except another book component (DocInfo, Glossary, and so on) or another section of the same or higher level; deeper sections must be nested properly.

```
<SECT1 ID=S1-061-ch05-1>
<TITLE>Running the Program
</TITLE>
<PARA>If you have the sample programs
and a workstation that runs X, you can try out this program
by compiling <FILENAME>basic/basicwin.c</FILENAME>.
```

```
</PARA>
</SECT1>
```

Sections have a Label attribute, in case you wish to hard-code section numbering for reference.

Chapter 4

Block-Oriented Elements

Most book components can contain the following block-oriented elements, some of them in formal and informal dress. Informal elements such as InformalTable are simply untitled equivalents of their formal sisters (Table), and are not to be listed in a LoT (such as a List of Tables).

- Highlights (list of main points)
- an Epigraph
- Paragraphs
- Lists
- Procedures
- MsgSets (error messages)
- Admonitions (Warnings, Tips, and so on)
- Tables and InformalTables
- Figures and Equations
- Examples and InformalExamples
- LiteralLayouts, ProgramListings, Screens, and ScreenShots
- Graphics
- BlockQuotes
- Sidebars

- Footnotes
- Admonitions, and other blocks

4.1 Highlights and Epigraph

Highlights (a section of major points appearing in the book component) typically appears below the Title or below any Epigraph, or may be extracted and presented separately.

An Epigraph (a section of poetry or prose, usually specially formatted) typically occurs between a book component's Title and other content.

4.2 Paragraphs

Paragraphs are the lowest level of ordinary text that you need to tag except for in-line elements. The DocBook DTD offers three kinds of paragraphs. Para may contain block-oriented elements (for example, lists, and Figures), while SimPara excludes them. FormalPara requires a Title, while Para and SimPara may not have Titles. FormalPara, Para, and SimPara are allowed in exactly the same contexts. No provision is made, in the case of SimPara, for indicating indentation of "continuation paragraphs", or paragraphs following a object such as a Figure; in the case of FormalPara and Para no such provision is needed.

In some cases, such as admonitions and Footnotes, paragraph tags are required within the outer tags, to allow multiple paragraphs. Thus a paragraph may be nested within a paragraph if a wrapper intervenes.

```
<WARNING>
<PARA>Never use software from sources
you would not trust with your firstborn.
</PARA>
<PARA>If in doubt about a program, don't use it at all.
</PARA>
</WARNING>
```

Lists contain ListItems, which typically contain paragraphs; ListItems, too, may contain more than one paragraph. (See List examples below.) Aside from straight text and, in the case of SimPara, block-oriented elements, paragraphs may contain any of the in-line elements discussed in the "In-line Elements" section.

4.3 Lists

There are six kinds of lists:

- ItemizedList (bulletted, dashed, and so on)
- OrderedList (numbered or lettered)
- VariableList (composed of paragraphs or sets of paragraphs with associated Terms)
- SegmentedList (a series of entries composed of equal numbers of segments, with optional titles pertaining to all the first segments, all the second segments, and so on)
- SimpleList (composed of terms that may be arranged in columns or in-line)
- GlossList (composed of GlossEntries only)

Lists may be nested within ListItems (an ItemizedList inside a ListItem of an OrderedList, for example), and ListItems may include paragraphs and other block-oriented elements which themselves may contain lists.

ItemizedLists use the Mark attribute to determine the character that precedes each ListItem. You may leave this to an application to decide, or supply a text string (there is no default and there are no generally agreed upon strings) to specify the mark that should appear before each ListItem. While the value of the Mark attribute is not confined to an entity (such as ✓) the processsing expectation should be that it is. The Mark value can be overridden at the ListItem level by the value of the ListItem's Mark attribute, if it is supplied.

Here's an ItemizedList:

```
<ITEMIZEDLIST>
<LISTITEM>
<PARA>This function should connect the client to an X server with
<FUNCTION>XOpenDisplay</FUNCTION>,
and exit gracefully if the connection could not be made.
</PARA>
</LISTITEM>
```

```
<LISTITEM>
<PARA>This para discusses
information about the physical screen.
</PARA>
<PARA>This is a second paragraph occurring
within the same ListItem.
</PARA>
</LISTITEM>
</ITEMIZEDLIST>
```

Normally an ItemizedList nested within a list of the same type is given a different mark or alphanumeric style (as in an outline). The DocBook DTD does not define any default sequence of marks. You must either indicate them by using the appropriate attributes or instruct your rendering application that, for example, the sequence of marks for an ItemizedList that contains another ItemizedList is bullets, then dashes.

OrderedLists may use the Numeration attribute to determine how the list is numbered. This attribute has a delimited set of values: Arabic (1, 2, 3), Upperalpha (A, B, C), Loweralpha (a, b, c), Upperroman (I, II, III), or Lowerroman (i, ii, iii). If extensions are needed, for example in the case of Japanese or Arabic documents, the values LocalAlpha and LocalNumber might be used, though these are not now supported. There is no default. OrderedList also has a Continuation attribute, to indicate whether the numbering of a list begins afresh (the default) or continues the numbering of the immediately preceding list. Finally, OrderedList has an InheritNum attribute, the values of which are Inherit and Ignore. Ignore is the default; i.e., numbering starts over (1, 2, 3 instead of 3.1, 3.2, 3.3). Set InheritNum to Inherit if you want nested lists to pick up the numbering of the ListItem within which they are nested (an OrderedList nested under the third ListItem of another OrderedList is numbered 3.1, 3.2, 3.3, instead of 1, 2, 3).

```
<ORDEREDLIST>
<LISTITEM>
<PARA>The window is mapped with <FUNCTION>XMapWindow</FUNCTION>
or related routines.
</PARA>
</LISTITEM>
<LISTITEM>
<PARA>All its ancestors must be mapped.  This
condition is always satisfied for the children of the root
window, the top-level windows of each application.
</PARA>
</LISTITEM>
<LISTITEM>
```

```
<PARA>The window must not be obscured by visible
sibling windows or their
ancestors this depends on the stacking order.  When first
mapped, a window appears on top of its siblings, which will
be on top of all windows if its parent is the root window.
</PARA>
</LISTITEM>
</ORDEREDLIST>
```

A VariableList may have a Title and TitleAbbrev, and must have VarListEntries, each of which contains a set of one or more Terms followed by a ListItem. As of this revision, the set of Terms (even if there is only one Term) must be contained with a TermGrp wrapper, but as both start- and end-tags of TermGrp are omissible, instances conforming to DocBook 2.1 need not be changed and TermGrp may be omitted when constructing instances. (However, once normalized, an instance will probably contain TermGrp tags.) Terms may contain in-line elements. Schematically a VariableList looks like this:

```
VariableList
     VarListEntry
  Term
  Term
  ListItem
       Para
       Para
     VarListEntry
  Term
  ListItem
       Para
     VarListEntry
  Term
  ListItem
       Para
```

In actual markup:

```
<VARIABLELIST>
<VARLISTENTRY>
<TERM>Window
</TERM>
<LISTITEM>
<PARA>A unique integer identifier (ID) that is
returned by XCreateWindow or XCreateSimpleWindow and is
```

```
thereafter used by the program to refer to the created
window resource.
</PARA>
</LISTITEM>
</VARLISTENTRY>
<VARLISTENTRY>
<TERMGRP>
<TERM>Display
</TERM>
<TERM>XOpenDisplay
</TERM>
</TERMGRP>
<LISTITEM>
<PARA>A large structure that contains information
about the server and its screens.  It is filled only after
this program connects to a server by calling XOpenDisplay.
</PARA>
<PARA>A small structure containing information about
the server and its screens would be more economical, but we
mention it only to generate another paragraph.
</PARA>
</LISTITEM>
</VARLISTENTRY>
</VARIABLELIST>
```

A SegmentedList is composed of labelled sets of information, and may be used to represent information that is often presented in simple tables. A SegmentedList may have a Title and TitleAbbrev, followed by any number of SegTitles and two or more SegListItems. SegListItems are composed of two or more Segs, which may contain in-line elements. You might render a SegmentedList in tabular form, or use some other typographic arrangment, such as:

```
<SEGMENTEDLIST>
<SEGTITLE>Museum
</SEGTITLE>
<SEGTITLE>City
</SEGTITLE>
<SEGTITLE>Collections
</SEGTITLE>
<SEGLISTITEM>
<SEG>Asian Art Museum
</SEG>
<SEG>San Francisco
</SEG>
<SEG>East Asian, Indian, and Islamic
```

```
</SEG>
</SEGLISTITEM>
<SEGLISTITEM>
<SEG>British Museum
</SEG>
<SEG>London
</SEG>
<SEG><EMPHASIS>all areas</EMPHASIS>
</SEG>
</SEGLISTITEM>
<SEGLISTITEM>
<SEG>Louvre
</SEG>
<SEG>Paris
</SEG>
<SEG>mostly paintings
</SEG>
</SEGLISTITEM>
</SEGMENTEDLIST>
```

This SegmentedList might appear in print as follows:

```
Museum:  Asian Art Museum
City:  San Francisco
Collections:  East Asian, Indian, and Islamic


Museum:  British Museum
City:  London
Collections:  all areas


Museum:  Louvre
City:  Paris
Collections:  mostly paintings
```

A SimpleList is intended for long lists of single words or short phrases. It consists of one or more Members, and has Columns and Type attributes. The value of Type may be Vert (the default), Horiz, or Inline. Inline indicates that the list should be formatted as a regular part of a paragraph, with commas and "and" inserted, such as:

```
<simplelist columns=4>
<member>a</member>
```

```
<member>b</member>
<member>c</member>
<member>d</member>
<member>e</member>
<member>f</member>
<member>g</member>
<member>h</member>
<member>i</member>
<member>j</member>
<member>k</member>
<member>l</member>
</simplelist>
```

`a, b, c, d, e, f, g, h, i, j, k,` and `l.` Vert and Horiz indicate that the Members should be displayed in the number of columns specified by the Columns attribute (the default is 1). If the value of Type is Vert, the Members should be ordered from top to bottom of each column:

```
a   d   g   j
b   e   h   k
c   f   i   l
```

If Horiz, the Members should be ordered left to right, one row at a time:

```
a   b   c   d
e   f   g   h
i   j   k   l
```

4.4 Procedures

A Procedure is a quasilist of operations to be performed. Procedure may have a Title and TitleAbbrev, followed by block-oriented elements, such as paragraphs, followed by one or more Steps. A Step may have a SubSteps wrapper for Steps nested within it, and this nesting may continue indefinitely (contrast the methods of nesting lists and Sects). This construction is intended to maximize the reusability of subsections of Procedures. Note that "SubSteps" is an element name, not the plural of SubStep (there is no element named SubStep).

After an optional Title, a Step must consist either of block-oriented elements such as paragraphs followed optionally by a SubSteps element, or simply a SubSteps element. A SubSteps then contains one or more Steps—it is a wrapper. Block-oriented elements may occur between the SubStepses.

```
<PROCEDURE>
<TITLE>Waking Up</TITLE>
<PARA>This is what you must do to awaken.
</PARA>
<STEP>
<PARA>Bring yourself to a hypnopompic state, either from an
ongoing dream or by use of your internal clock.  You may
feel unable to move, but you will no longer be dreaming.
</PARA>
<PARA>Now you are ready for real-world readjustment.
</PARA>
<SUBSTEPS>
<STEP>
<PARA>Roll over.
</PARA>
</STEP>
<STEP>
<PARA>Squint out of one eye.
</PARA>
</STEP>
</SUBSTEPS>
</STEP>
<STEP>
<PARA>Yawn and rise from your bed.
</PARA>
</STEP>
</PROCEDURE>
```

4.5 MsgSet

A MsgSet is a quasilist of messages produced by a system, with various additional information. These are typically error messages, but they could conceivably be congratulatory or soothing messages to the user. Please note that the entire Msg* construction is new and complicated; it may be simplified in the future.

MsgSet contains one or more MsgEntries, which are merely wrappers, each containing one or more Msgs, followed by an optional MsgInfo, then any number of MsgExplans.

Msg is the part of a MsgEntry that contains the error message and its subparts, along with explanatory text. A Msg has a required MsgMain, followed by any number of MsgSubs and MsgRels, in any order.

MsgMain is the main error message of a Msg. MsgMain begins with an optional Title, followed by MsgText, the real content of the message (containing may be block-oriented elements).

MsgSub contains messages that appear in various contexts. It, too, contains a an optional Title followed by MsgText. MsgRel contains a message that is related to the main message (MsgMain) but which appears in a different place. For example, MsgMain might be a message that appears on a network client, and MsgRel a related message that appears at the server console in response to the same condition or event. MsgRel begins with an optional Title and contains MsgText.

That finishes the Msg part of MsgEntry; there remain MsgInfo and MsgExplan.

MsgInfo holds information about the Msg containing it. It may have any number of MsgLevels, MsgOrigs, and MsgAuds, in any order. MsgLevel describes, in plain text, the level of importance or severity of a Msg. MsgOrig describes, in plain text, the origin of a Msg. MsgAud describes, again in plain text, the audience to which a Msg is relevant.

MsgExplan is a holder for any kind of explanatory material relating to the Msg. MsgExplan begins with an optional Title (typically something such as "Explanation:" or "Action:") and may contain block-oriented elements.

```
<MSGENTRY>
<MSG>
<MSGMAIN>
<MSGTEXT>
<PARA>Auto removal: Token-ring lobe
</PARA>
</MSGTEXT>
</MSGMAIN>
<MSGSUB>
<TITLE>Failure
</TITLE>
<MSGTEXT>
<PARA>Local token-ring adapter
</PARA>
<PARA>Local access unit
</PARA>
<PARA>Local lobe cables
</PARA>
</MSGTEXT>
</MSGSUB>
<MSGREL>
```

```
<TITLE>NetWare message:
</TITLE>
<MSGTEXT>
<PARA>TOKEN-NW-237-Adapter <REPLACEABLE>number</REPLACEABLE&
gt; -board <REPLACEABLE>number</REPLACEABLE> : The adapter
is beaconing.
</PARA>
<PARA>Ring status = <REPLACEABLE>code</REPLACEABLE>
 .
</PARA>
</MSGTEXT>
</MSGREL>
     </MSG>
<MSGINFO>
<MSGORIG>NetView for NetWare
</MSGORIG>
<MSGLEVEL>Informative
</MSGLEVEL>
     </MSGINFO>
<MSGEXPLAN>
<TITLE>Explanation:
</TITLE>
<PARA>Adapter left the ring as part of the beacon automatic
recovery process.
</PARA>
</MSGEXPLAN>
<MSGEXPLAN>
<TITLE>Action:
</TITLE>
<PARA>Ask the LAN administrator to review the link data.
</PARA>
</MSGEXPLAN>
</MSGENTRY>
```

4.6 Tables

A Table is an ordered array of text, usually with a title and headings. This information may be represented in SGML, but it may also be convenient to provide it in another form. The DocBook DTD allows you to mark up a table in SGML or to provide a graphic representation of it in an external file by using the Graphic element. The Table fragment was derived from the CALS MIL-28001 DTD.

A Table has a required Title, a TitleAbbrev, and one or more Graphics (which may be pointers to external files that contain the Table formatted in a different way) or one or more TGroups, which are slabs of CALS table. For details on the contents of Table see the alphabetical list of elements; Tables may not be nested within Tables, but a Table may contain an EntryTbl, which achieves the same effect. InformalTable is Table's untitled sister.

With this revision, the content model of Entry (the contents of a table cell) has been broken into two possibilities: paragraph-like elements (Para, Note, LiteralLayout, lists, etc.), and in-line content, to avoid the unwanted possibility that block-oriented elements could be mixed with free text. You may include one but not the other. However, this means that no line breaks may occur between tags within an Entry (the only place in this DTD where this effect is produced). In the future, a wrapper for in-line content may be provided so as to avoid this difficulty.

```
<TABLE>
<TITLE>Sample Decimal-Aligned Table
</TITLE>
<TGROUP ALIGN="char" CHAROFF="50" CHAR="." COLS="3">
<TBODY>
<ROW>
<ENTRY>345.021</ENTRY>
<ENTRY>211.02</ENTRY>
<ENTRY>221.3</ENTRY>
</ROW>
<ROW>
<ENTRY>blue</ENTRY>
<ENTRY>red</ENTRY>
<ENTRY>yellow</ENTRY>
</ROW>
</TBODY>
</TGROUP>
</TABLE>
```

For Graphic, see *Figure,* below.

4.7 Figures and Equations

A Figure must have a Title, may have a TitleAbbrev, and may contain one or more of the following: BlockQuote, CmdSynopsis, FuncSynopsis, InformalEquation, Graphic, InformalTable, Link, LiteralLayout, OLink, ProgramListing, Screen, ScreenShot, Synopses, and ULink.

A Graphic may contain graphical data or point to an exterior file that contains such data. It has an ID attribute and a Format attribute that indicates how the data is formatted. Format may have the value CGM-BINARY, CGM-CHAR, CGM-CLEAR, DITROFF, DVI, EPS, EQN, FAX, GIF, IGES, PIC, PS, TBL, TEX, TIFF (see the "NOTATIONS" section of the DTD for details about these formats). There is no default. Graphic may point to an external file via its Fileref or Entityref attributes. The value of Fileref should be a filename; the value of Entityref should be that of an external data entity. These attributes may conflict, so an order of precedence is specified: if data is given as the content of Graphic, neither Entityref nor Fileref should be given but a Format value should be supplied. If no data is given as the content of Graphic, only one of Entityref Fileref should be given; if both are present, Entityref preëmpts Fileref. If either of those attributes is used, Format should not be.

Whether Graphic occurs within or between paragraphs, it is to be displayed as a block (that is, Graphic is an informal figure); if a graphical image is to be presented in-line, it must be marked as an InlineGraphic, which is otherwise just like Graphic.

Figure has common, Label, and Float attributes; Float indicates whether the Figure is to be rendered where convenient (use the value 1) or at the place it occurs in the text (the default; use the value 0).

```
<FIGURE ID="FG-061-ch05-2" FLOAT="1">
<TITLE>The xdmlogwin Display
</TITLE>
<GRAPHIC FILEREF="gr061ch05t2.gif"></>
<GRAPHIC ENTITYREF="&gr061ch05t3.gif;"></>
</FIGURE>
```

Equations are solely graphical (math support is to be added in a future revision). An Equation may have a Title and TitleAbbrev, and then contains one or more InformalEquations or Graphics, in any order. InformalEquation is an untitled equation to be displayed as a block, like InformalTable, and may contain one or more Graphics; it has common attributes. InlineEquation is just the same as InformalEquation, but is to be display in-line.

4.8 Examples and InformalExamples

Example is defined very loosely, so that a program listing can be accompanied by other block-oriented elements. An Example must have a Title, may have a TitleAbbrev, and then contains any combination of paragraphs, lists, and other block-oriented elements. An InformalExample is just the same, but without a Title.

4.9 LiteralLayouts, ProgramListings, Screens, and ScreenShots

The DocBook DTD provides three elements for material set off from the main text, in which white space and line breaks are to be considered significant: LiteralLayout, ProgramListing, and Screen. All of these elements may contain in-line elements only, not block-oriented elements. They all have a Width attribute, which is some number indicating to an application how wide the widest line is, and a fixed Format attribute, with the value "linespecific."

LiteralLayout is the simplest of the linespecific elements. It consists only of the set-off lines:

```
<literallayout>
Here's a line
here's another, below it
   and a third line, indented
</literallayout>
```

A ProgramListing is just like a LiteralLayout, but should be reserved for listings of programs, which may require a different style of presentation. ProgramListings, like the other line-specific elements, may have LineAnnotations, which are a document author's comments on the code, not the comments written into the code itself by the code's author.

A Screen is a representation of all or part of a screen display, which may be textual or graphical. A Screen contains text and in-line elements, in the manner of a LiteralLayout. A Screen has no Title; if you want a Screen to have a Title, enclose it in a Figure or Example. Here's a Screen with text content:

```
<SCREEN ID="SC-1015-12-24">
<SYSTEMITEM CLASS="PROMPT">%</SYSTEMITEM> <USERINPUT>make install</USERINPUT>
```

```
install -c -m 0644 rgb.txt /usr/lib/X11
install -c -m 0644 rgb.dir /usr/lib/X11
install -c -m 0644 rgb.pag /usr/lib/X11
install -c -s showrgb /usr/lib/X11
install in ./rgb done
</SCREEN>
```

If you want to represent a Screen's contents graphically, use the ScreenShot element. A ScreenShot contains an optional ScreenInfo and a required Graphic. The ScreenInfo is a comment about how the image in the Graphic was created (so you can recreate it if need be), and is not meant to be displayed or printed; it contains plain text.

Here's a ScreenShot:

```
<SCREENSHOT>
<SCREENINFO>Press the F1 key and type &ldquo;catfile&rdquo;
while holding down the Alt, Page Down, and F12 keys.
</SCREENINFO>
<GRAPHIC FORMAT="GIF" FILEREF="figs/05.07.gif"></GRAPHIC>
</SCREENSHOT>
```

4.10 BlockQuotes

BlockQuote is intended for a quotation set off from the main text, rather than occurring in-line (for an in-line quote, use Quote). In a BlockQuote white space is not significant: the lines wrap. A BlockQuote contains paragraphs and other block-oriented elements, not plain text.

4.11 Sidebars

A Sidebar is a segment of text isolated from the narrative flow of the main text, typically boxed and allowed to float in the page layout. Magazines and newspapers contain many sidebars. A Sidebar may have a Title and a TitleAbbrev, and must have text in the form of paragraphs, lists, Figures, or other block-oriented elements; no Sects or other Sidebars are allowed.

4.12 Footnotes

Footnotes may be marked up in two ways. In one method, within a paragraph or other element, the location of the footnote mark should be indicated with FootnoteRef, which points to the corresponding Footnote with its Linkend attribute, the value of which is the ID of the associated Footnote. The Footnote need not be anywhere close to the FootnoteRef. FootnoteRef may contain plain text, which is the mark to be displayed, or it may be empty, in which case its Mark attribute provides another way of indicating the contents of the mark, such as an asterisk (*), a number (84), or a dingbat specified by a name that is to be interpreted by the application. While the DTD does not specify that the value of Mark be an entity, the processing expectation should be that it is (e.g., †).

Using the other method, a Footnote may occur at the spot where it is desired to generate the footnote mark. The mark may be left to be generated automatically, or a FootnoteRef may be supplied to specify it.

Footnote may contain any of the following, but may not contain another Footnote: BlockQuote, CmdSynopsis, FuncSynopsis, Graphic, InformalEquation, InformalTable, LiteralLayout, any kind of paragraph, ProgramListing, Screen, ScreenShot, Synopsis, and any kind of list.

4.13 Admonitions and Other Blocks

Admonitions are special sections of text that call attention to some matter of importance. Admonitions are one of the following elements: Caution, Important, Warning, Tip, and Note. They may have Titles, paragraphs, and other block-oriented elements, but no Sects or other admonitions.

Abstract (a document summary) and AuthorBlurb (author information) may have Titles and paragraphs.

Chapter 5

Elements Indicating Nonlinear Relationships

IndexTerms, Links, Anchors, and XRefs all indicate nonlinear relationships within the text—unlike the hierarchical nesting of Sects and the linear flow of paragraphs within Sects.

5.1 IndexTerms

IndexTerms are words or phrases to be indexed. The DocBook DTD provides for three levels of indexing, as well as for “see” and “see also” indexing. IndexTerms may occur almost anywhere, but their contents may not do double duty as part of the text: the contents of IndexTerm do not appear in the text itself.

An IndexTerm must contain a Primary, and may contain a See and one or more SeeAlsos, as well as a Secondary; within Secondary there may be a See and one or more SeeAlsos, as well as a Tertiary; a Tertiary may include a See and one or more SeeAlsos, also. IndexTerm has common, SpanEnd, PageNum, Significance, Scope, and Zone attributes.

The SpanEnd attribute is for use in an IndexTerm ending a span of text to be indexed; that IndexTerm must be empty (have no content or end tag) and must have as the value of SpanEnd the ID of the IndexTerm that begins the span, earlier in the text. The PageNum attribute may be used to indicate the page on which the indexed term is found in print. Significance may have the value Preferred, indicating that the entry is the most pertinent of the series, or

Normal (the default).

The Scope attribute, which may have the values All, Global, or Local (the default is IMPLIED), can be used to meet the case of IndexTerms that should be used to generate, e.g., an index for a Book (Local) but not an index for a Set (Global) or vice versa; the value All should be used when the IndexTerm should appear in all generated indices. You should define your implementation's default accordingly.

The Zone attribute is to be used when it is desired to attach indexing information to SGML elements rather than, or in addition to, placing IndexTerms in the flow of the text so as to be sure page references are generated correctly. Zone could be used to attach keywords to sections and paragraphs, for example, facilitating the assembly of nonstructural views of the indexed text. The values of Zone should be the IDs of the element or elements referred to.

```
<PARA>This paragraph deals with two subjects that should
be listed in
the index:  how to rotate your terminal and how to adjust
its height.
<INDEXTERM>
<PRIMARY>rotating your terminal
</PRIMARY>
</INDEXTERM>
<INDEXTERM>
<PRIMARY>terminal
</PRIMARY>
<SECONDARY>rotation of
</SECONDARY>
</PRIMARY>
</INDEXTERM>
<INDEXTERM>
<PRIMARY>terminal
</PRIMARY>
<SECONDARY>adjustment of
</SECONDARY>
<SEEALSO>troubleshooting
</SEEALSO>
</PRIMARY>
</INDEXTERM>
</PARA>
```

5.2 IndexEntries

IndexEntries, which occur in the Index, might be constructed by extracting and processing the IndexTerms.

IndexEntries themselves are put together just like IndexTerms, with the corresponding elements PrimaryIE, SecondaryIE, TertiaryIE, SeeIE, and SeeAlsoIE.

5.3 Links and Anchors

A link is a pointer to another section in the document, or to another document. The link elements are still somewhat provisional, and are under active review. The most conventional of them is Link, which may include in-line elements as text and has a required Linkend attribute, which is the ID of the element it is linked to. It also has a Type attribute, so you can assign a role to a Link. And Link has an Endterm attribute, the value of which may be the ID of an element whose content is to appear as the text of the Link. For example, Link might give the label of a hot spot explicitly:

```
<PARA>To go there,
<LINK LINKEND="H1-061-ch05-1">click here</LINK> and say goodbye.
</PARA>
```

or a Link might point to a section and display its title:

```
<PARA>Click to go to
<LINK LINKEND="S1-061-ch05-1" ENDTERM="T1-061-ch05-1"></LINK>
</PARA>
```

where the content of the element that bears the ID T1-061-ch05-1 (perhaps the title of Chapter Five) is displayed as the hot spot. (This is effectively an unclean form of CONREF.)

Please note that the common attribute XRefLabel provides another mechanism for displaying text associated with a Link's target.

Link uses the SGML mechanism of IDREF in pointing to an element by its ID. An SGML application may report an IDREF error when the ID does not occur in the document set being processed. This may be harmless.

Anchor is an in-line element that marks a location as a target for a Link (any element bearing an ID may be a target, of course). Anchor has ID, PageNum, Remap, Role, and XRefLabel attributes, but only the ID is required. It is an empty element: at minimum, the start tag is supplied, with an ID only:

```
<PARA>There is an Anchor <ANCHOR ID="ch05-061-AN-7"> in this sentence.
</PARA>
```

You could also supply a value for the XrefLabel attribute to provide text to be generated by an XRef to the point marked by that Anchor.

Two other links are provided, OLink (a link that performs some operation on its target) and ULink (a link that uses a Universal Resource Locator, useful for making reference by pathname and filename).

Neither OLink nor ULink is a traditional SGML link, though both are fully conformant to ISO 8879 (the SGML standard). In particular, ULink stores all link information in a format that SGML does not process. Conventional SGML browsers may not yet work with ULinks.

5.4 Cross References (XRefs)

An XRef is a cross reference to another part of a document. It has Linkend and Endterm attributes, just like Link, but like Anchor, it may not have content. XRef must have a Linkend, but the Endterm is optional. If it is used, the content of the element it points to is displayed as the text of the cross reference; if it is absent, the XRefLabel of the cross-referenced object is displayed.

```
<PARA>See <XREF LINKEND="ch05-s1" ENDTERM="ch05-t1">,
for more information.</PARA>
```

The example above references the section that has the ID "ch05-s1" and supplies the text of its Title, which has the ID "ch05-t1", in the manner of: "See *Terminal Emulation and the xterm Terminal Type* for more information."

To include in the cross reference text generated text that may be associated with the target, e.g., "Figure 5-1.", use the same mechanism you employ to generate that text in the first place (your application's defaults or style sheet).

Chapter 6

In-line Elements

In-line elements are used to label specific terms or usage. In-line elements may appear anywhere in a block-oriented element, but may not span two block-oriented elements. In documentation these elements are typically indicated by font changes.

All of the elements defined in the DTD in the parameter entities that comprise inlinechar.gp may appear in-line. (See the Reference section for brief definitions of in-line elements.)

Chapter 7

Using In-line Elements

Trying to determine the most specific in-line element that can be applied to the specialized terms in your document can be time-consuming and frustrating. Instead, you may wish to tag these terms using only the most general in-line elements. We have found it useful, in converting files from troff to SGML, to translate the font changes that mark off these terms in print into five DocBook tags: Emphasis, Literal, Parameter, UserInput, and Interface (which correspond well to our conventional use of italic, typewriter font, italic typewriter, bold typewriter, and sans serif; you may wish to choose other values).

Some such tags (Application, Database, Interface, MediaLabel, Parameter, Replaceable, SystemItem, Trademark, and SGMLtag) have a Class attribute with delimited values for each element. This mechanism provides a two-step approach toward specifying the meanings of in-line terms. For example, an Interface may be further specified as Button, Icon, Menu, or MenuItem through its Class attribute. You can thus indicate the general meaning of the term or be more precise, as you like.

Chapter 8

Articles

A Book may consist of Articles, in which case it represents an issue of a journal (and you should provide its ISSN, not ISBN, in the BookInfo) or a collective work, such as an anthology.

An Article has, in order, a required ArtHeader, main contents as for Chapter, then any number of Indexes, Glossaries, Bibliographies, Appendixes and Acknos, in any order. Article has common and ParentBook attributes, the latter pointing to the ID of the enclosing Book. For the details of ArtHeader, see the alphabetical list of elements.

Chapter 9

Reference Entries

A man page or reference page is a distinct form of reference information. We have modelled our RefEntry element on man pages. In this DTD, a RefEntry may occur within a Chapter or Appendix, or a collection of RefEntries may be made into a book component called Reference.

9.1 RefEntry Basics

A RefEntry contains, in order: one DocInfo (optional); one RefMeta (optional); one or more Comments, links, FootnoteRefs, or XRefs (optional); one RefNameDiv (required); one RefSynopsisDiv (optional); one or more RefSect1s (required). Schematically, with additional elements particular to RefEntries:

```
RefEntry
     DocInfo
     RefMeta
  RefEntryTitle
  ManVolNum
  RefMiscInfo
     RefNameDiv
  RefName
  RefPurpose
     RefSynopsisDiv
  Synopsis
     RefSect1
  RefSect2
```

```
RefSect2
   RefSect1
   RefSect1
```

9.2 RefEntry Variations

Reference entries have conventional divisions, but no single definitive conventional format. The arrangement of information in the initial divisions of a reference entry may be rather complicated. RefEntry accomodates several alternate structures for this information. For example, a typical reference entry has only a single subject (here, RefName), which is named in a division that is usually called "Name" (here, RefNameDiv), and is followed by an em dash and a short descriptive phrase (here, RefPurpose). A reference entry for *mwm,* the Motif window manager, might have this text in its Name division: "mwm — the Motif window manager"

This is simple to tag: "mwm" is a RefName and the descriptive phrase is a RefPurpose (the em dash should not be represented in the SGML document).

Other reference entries may provide a purpose statement on more than one subject (command, function, file format, and so on). For example, in a reference entry for *grep* there may be several subjects described by the same purpose statement: "grep, egrep, fgrep — search a file for a string or regular expression"

Here all the terms in the series are RefNames, and should be tagged individually.

Still other reference entries may provide a group name for a series of subjects that is not itself one of the subjects discussed. For example, in an AT&T reference entries called "string," the name of the reference entry is given in this manner: "string: strcasecmp, strncasecmp — string operators"

In this series, "string," is not a function name like the terms that follow the colon following "string." In a RefEntry the word "string" is tagged as a RefDescriptor, which is an optional rather than a required element of RefNameDiv.

Many of the higher-level elements of RefEntry may occur only within RefEntries, not within Chapters or other book components. These are: RefMeta, RefEntryTitle, ManVolNum, RefMiscInfo, RefNameDiv, RefDescriptor, RefName, RefPurpose, RefSynopsisDiv, and RefSect1–3.

9.3 RefMeta

The RefMeta contains, in order, a RefEntryTitle, an optional ManVolNum, and any number of RefMiscInfos.

All this is metainformation, some of which may occur in the heading of a reference entry. None of it is from the body of a reference entry. RefEntryTitle is the primary name by which the reference entry is sorted and indexed. ManVolNum is the alphanumerical designation often supplied in parentheses after the RefEntryTitle, as in "string(3C)." It refers to the historical arrangement of reference entries in sections. Other pieces of information may follow the ManVolNum, each captured with the tag RefMiscInfo, so that you may have a series of RefMiscInfos (as many as eight have been reported in legacy documents). For example, O'Reilly's reference entry for XmuReshapeWidget includes the heading "Xmu—Shape Extension Utilities," which should be tagged as a RefMiscInfo. Other examples of RefMiscInfo contents are copyright, date, draft status, hardware architecture, and a descriptive phrase for use in a print header. Of these RefMeta elements, only RefEntryTitle may contain in-line elements; the others may contain only plain text.

Do not be confused by the fact that RefEntryTitle may contain the same contents as RefName in the RefNameDiv. For the *mwm* reference entry, "mwm" is the content of both RefEntryTitle and RefName. This may not always be the case, as in the "string" example above: in that instance, "string" is the content of RefEntryTitle, but it is not a RefName. Remember, the RefMeta does not contain the contents of the Name division of a reference entry, only metainformation.

```
<REFMETA>
<REFENTRYTITLE>ldsysdump
</REFENTRYTITLE>
<MANVOLNUM>1
</MANVOLNUM>
<REFMISCINFO CLASS="Sysad">System Administration Commands
</REFMISCINFO>
<REFMISCINFO CLASS="Copyright">Copyright 1985
</REFMISCINFO>
</REFMETA>
```

9.4 RefNameDiv

RefNameDiv is the top division of a reference entry, the one usually labelled "Name." The RefNameDiv contains, in order, an optional RefDescriptor (for reference entries such as the "string" example), one or more RefNames, a required RefPurpose, followed by any number of RefClasses. Comments, FootnoteRefs, links, or XRefs may also appear at the end of RefNameDiv.

In RefClass you indicate the applicability or scope of the topic of a RefEntry (such as "LocalBlocking"). RefName may contain further nested elements from the parameter entity computerterms.gp, such as Command or Function. RefPurpose may contain any in-line elements.

The examples of RefNameDiv cited above are tagged like this:

```
<REFNAMEDIV ID="061-RP72">
<REFNAME>mwm
</REFNAME>
<REFPURPOSE>the Motif window manager
</REFPURPOSE>
</REFNAMEDIV>
```

```
<REFNAMEDIV ID="061-RP73">
<REFNAME>grep
</REFNAME>
<REFNAME>egrep
</REFNAME>
<REFNAME>fgrep
</REFNAME>
<REFPURPOSE>search a file for a string or regular expression
</REFPURPOSE>
</REFNAMEDIV>
```

```
<REFNAMEDIV ID="061-RP74">
<REFDESCRIPTOR>string
</REFDESCRIPTOR>
<REFNAME>strcat
</REFNAME>
<REFNAME>strdup
</REFNAME>
<REFNAME>strncat
</REFNAME>
<REFNAME>strcmp
```

```
</REFNAME>
<REFNAME>strncmp
</REFNAME>
<REFPURPOSE>string operators
</REFPURPOSE>
</REFNAMEDIV>
```

9.5 RefSynopsisDiv

RefSynopsisDiv is intended for the second division of reference entries, usually titled Synopsis or Syntax. But there are reference entries without Synopsis or Syntax divisions, so RefSynopsisDiv is an optional part of RefEntry. The content of this division is often simply a single line giving the syntax for the subject of the reference entry, or something less constrained, in some cases including subheadings. Thus the content of RefSynopsisDiv may be one or more synopses (Synopsis, CmdSynopsis, or FuncSynopsis, containing a single line or a set of lines), or one or more synopses followed by any number of RefSect2s, or one or more RefSect2s.

Here is a valid RefSynopsisDiv:

```
<REFSYNOPSISDIV ID="RS-061-66">
<SYNOPSIS>
<COMMAND>ls</COMMAND>
<OPTIONAL><PARAMETER>options</PARAMETER></OPTIONAL>
<REPLACEABLE CLASS="PARAMETER"><FILENAME>directory name
</FILENAME>
</REPLACEABLE>
</SYNOPSIS>
</REFSYNOPSISDIV>
```

9.6 Synopses

For explanations of the CmdSynopsis and FuncSynopsis elements, please see the alphabetical list of elements.

9.7 RefHeads

The rest of a reference entry consists of one or more divisions, which are to be tagged as RefSect1s, which may have RefSect2s, which may have RefSect3s. These are just like Sect1–3, with the same required and allowable contents.

Chapter 10

Miscellaneous

Some elements are not part of the illustrations or body of text in a printed work. Such elements may appear in a printed work but are not part of the text. Others are provided in case they should be needed for some aspect of on-screen display or book production (TitleAbbrevs). Still others are metainformation, such as comments.

TitleAbbrev may be provided where a long Title might be inappropriately truncated in some on-line display such as a title bar, or for use in a printed header or footer.

RevHistory, an element of DocInfo, records the authorship of changes to a book. It consists of one or more Revisions, each of which must contain a RevNumber and a Date, and may have one or more AuthorInitials and a RevRemark.

A Comment, containing in-line elements and plain text, is something not meant to be displayed or printed, such as a message to a production editor. Beware! Comment is not an SGML comment, it is an element. Just tagging text as a Comment will not ensure that it is hidden; you must arrange that in your application. And in line-specific elements you may find that the line breaks in your comment are rendered by your application.

Chapter 11

The Elements Alphabetized

Emphasized entries indicate block-oriented elements.

11.1 Common Attributes

Common attributes are ID, Lang (language), Remap (to store the former GI of an element in a document filtered to Docbook), Role (to type Docbook elements when need be), and XRefLabel (text to be displayed as the label of cross references to a block-oriented element).

ID is an identifier, which must be a string that is unique at least within the document and which must begin with a letter. Lang should be a language code drawn from ISO 639 (perhaps extended with a country code drawn from ISO 3166, as en_US). Use it when you need to signal your application to change hyphenation and other display characteristics. Remap is a device for preserving a previous tag name when fitting a document to the DocBook DTD. Think of ¡CHAPTER Remap="Fascicle"¿ as saying, "I'm a Chapter now, but in a previous life I was a Fascicle." Role allows you to clone DocBook elements without creating new ones: if you have lists of vegetables and lists of fruit (perhaps displayed differently), give them appropriate Role attributes, such as ¡TERM Role=vegetables¿ and ¡TERM Role=fruit¿.

11.2 Other Attributes

Certain other attributes occur regularly. PageNum is the number of the page on which a given element begins or occurs in a printed book. Label holds some text associated with its element that is to be output when the document is rendered. XRefLabel holds some text that is to be used when a cross reference (XRef) is made to its element (XRefLabel may become an element of its own in the future). Type is used with links, as it is clear that different types of links may be required; it duplicates the function of Role.

Elements that bear the Class attribute, such as Interface, have general meanings that can be made more specific by providing a value for Class from the delimited list for that element. For example, for the Interface element one may specify Menu, or Button; for the MediaLabel element one may specify CDRom or Tape. Each element has its own list of permissible values for Class, and no default is set, so you can ignore this attribute if you wish.

Some attributes are defined with the keyword %yesorno;, which resolves to NUMBER. The number must be either 1 (yes) or 0 (no).

An attribute that has the keyword IMPLIED bears no processing expections if it is absent or its value is null. Application designers might wish to supply plausible defaults, but none is specified here. However, it would be plausible to imply that the language of a DocBook instance is U.S. English unless specified otherwise.

11.3 "In-line" vs. "In flow"

In this document, "in-line" means "occuring within a line of text, like a character or character string, not causing a line break." This term is sometimes used to refer to objects such as an illustration around which something like a paragraph is wrapped; here that circumstance will be called "in flow." There is no provision yet for indicating that an object is in flow, but one could make creative use of the Role attribute to do so.

A related point: formal objects have titles; informal objects do not. That an object is informal does not mean that it is in-line: these are two different characteristics.

11.4 CALS Tables

The Table model used here is derived from MIL-M-28001B, dated 26 July 1993 (supersedes MIL-M-28001A of 20 July 1990.) The documentation of Table and its subelements, below, has been taken from that document, with some editing and attempts at clarification.

11.5 List of Elements

Abstract A document summary. Abstract contains an optional Title followed by paragraphs, and has common attributes.

Abbrev An abbreviation, especially one followed by a period. It contains plain text and has common attributes.

Ackno Acknowledgements in an Article. It contains plain text and has common attributes.

Acronym A pronounceable contraction of initials, usually printed in all caps or small caps. It contains plain text and has common attributes.

Action A function invoked in response to a user event. It may contain members of cptrphrase.gp, and has common and MoreInfo attributes. For the MoreInfo attribute see *Application.*

Address A real-world address. It contains any number, in any order, of: Street, POB, Postcode, City, State, Country, Phone, Fax, Email, and OtherAddr. It has common attributes.

Affiliation Author's institutional affiliation. It contains, in order, an optional ShortAffil, any number of JobTitles, optional OrgName, any number of Orgdivs, and any number of Addresses. It has common attributes.

Anchor Marks a target for a Link. Anchor may appear almost anywhere, and has no content. Anchor has optional ID, Pagenum, Remap, Role, and XRefLabel attributes.

Appendix May occur only after Chapters or References, in a Book. Appendix begins with an optional DocInfo, followed by a required Title, optional TitleAbbrev, and anything found in the body of a Chapter. It has common and Label attributes.

Application The name of a software program. It may contain in-line elements and plain text, and has common, Class, and MoreInfo attributes. Class may be Hardware or Software (no default). MoreInfo may have the value RefEntry, indicating that a RefEntry exists containing additional information about this term. The default value for MoreInfo is None.

Arg Argument in a CmdSynopsis. Arg may contain any number of Args, Groups, Options, SynopFragmentRefs, Replaceables, in any order, mixed with plain text. Arg is defined in the DTD as having common attributes and the attributes defined by the parameter entities argchcatt and repatt.

argchcatt or "argument choice attribute" resolves to the attribute Choice, with allowed values Opt (the Arg is optional; the default), Req (it is required), and Plain (neither optional nor required).

repatt or "repetition attribute" resolves to the attribute Rep, with allowed values Repeat (the Arg may be repeated; the default) and Norepeat (the Arg does not repeat).

ArtHeader Metainformation for an Article. ArtHeader has, in order, a required Title, optional TitleAbbrev, optional Subtitle, one or more AuthorGroups, optional BookBiblio, a required ArtPageNums, any number of Abstracts, any number of ConfGroups, and finally any number ContractNums and ContractSponsors, in any order. ArtHeader has common attributes.

Article An article in a journal. Book may be used to hold Articles. An Article has, in order, a required ArtHeader, main contents as for Chapter, then any number of Indexes, Glossaries, Bibliographies, Appendixes and Acknos, in any order. Article has common and ParentBook attributes, the latter pointing to the ID of the enclosing Book.

ArtPageNums Page numbers of an Article as published. It contains plain text (e.g., 23–147) and has common attributes.

Author Author of a document, occuring in AuthorGroup. It consists of one or more of the following, in any order: Honorific, Firstname, Surname, Lineage, OtherName, Affiliation, Contrib, and AuthorBlurb. It has common attributes.

AuthorBlurb Short description of author. AuthorBlurb contains an optional Title followed by paragraphs, and has common attributes.

AuthorGroup Wrapper for Author information. It contains one or more CorpAuthors, Collabs, Editors, OtherCredits, and Authors, in any order. It has common attributes.

AuthorInitials Indicates the author of a Revision or Comment. It contains plain text and has common attributes.

BeginPage BeginPage marks a page break in a print version of a work that may be displayed online. It is without content, but has optional ID, PageNum, Remap, Role, and XRefLabel attributes. The value of PageNum, which is not required, should be the folio (page number) of the page beginning at that point. *Usage Note:* Once you give a PageNum to a BeginPage, subsequent BeginPages should be assumed to indicate that value incremented by one per BeginPage. You can indicate a change or reset in page numbering by providing a PageNum value for a later BeginPage.

BeginPage, like IndexTerm, is handled as an SGML inclusion. If you parse a DocBook instance that has as its doctype a structure smaller than Sect1, your SGML parser will not pick up the inclusion, and you will have to include a modification of the doctype element in the document type declaration, or redefine the element in your local copy of the DTD.

BiblioDiv A section of a Bibliography. It may have, in order, an optional Title, optional TitleAbbrev, any number of block-oriented elements, followed by one or more BiblioEntries. It has common attributes.

BiblioEntry An entry in a Bibliography. It may begin and end with BiblioMiscs, between which must be an ArtHeader, BookBiblio, or SeriesInfo. That is, the main content of a BiblioEntry may be identical to a section of metainformation. BiblioEntry has common attributes.

Bibliography A bibliography. It may be a book component on its own, or may appear within a Preface, Chapter, or Appendix, or at the end of a Glossary. It may have a DocInfo, a Title and a TitleAbbrev, then optional block-oriented elements, and then one or more BiblioEntries or one or more BiblioDivs. It has common attributes.

BiblioMisc Untyped information required in a BiblioEntry or BookInfo, in addition to the elements required there. It contains plain text and has common attributes.

BlockQuote A quotation set off from the main text, rather than occurring in-line. It may have a Title, followed by block-oriented elements. It has common attributes.

Book A collection of book components. The Book content model is loose enough to accomodate English, French, and Japanese books. A Book may have a Title and TitleAbbrev, followed in order by an optional BookInfo, an optional ToC, any number of LoTs, any number of Prefaces, main contents, and back matter. The main contents are required, and must be one or more Parts; one or more Chapters followed by any number of References; one or more Articles; or one or more References. All back matter is optional, but must appear in order: any number of Appendices, a Glossary, a Bibliography, and any number of Indexes and SetIndexes, followed by any number of LoTs followed by an optional ToC. Book has common, FPI, and Label attributes. The FPI attribute is intended to hold an SGML Formal Public Identifier for the Book; the Label attribute can be used to supply the number of a Book, or you can use the content of the VolumeNum element in BookInfo. (In constructing Formal Public Identifiers you may wish to use your ISBN publisher's prefix. In the ISBN 1-565692-043-0, for example, the publisher's prefix is 1-565692.)

BookBiblio All the information about a book that may be relevant to a bibliographical citation; occurs in BookInfo and may be used in BiblioEntry. Only Title and AuthorGroup are required. BookBiblio may contain, in order: the required Title, optional TitleAbbrev, Subtitle, and Edition, followed one or more required AuthorGroups; then, optionally, either an ISBN followed by an optional VolumeNum or an ISSN followed by optional VolumeNum, optional IssueNum, and an optional PageNums. After these elements there may occur, again optionally and in order, InvPartNumber, ProductNumber, ProductName, PubsNumber, and ReleaseInfo; then there may be any number of Pubdates followed by any number of Publishers, followed by optional Copyright, then optional SeriesInfo; then any number of Abstracts, any number of ConfGroups, any number of ContractNums mixed with any number of ContractSponsors, and an optional PrintHistory followed by an optional RevHistory. BookBiblio has common attributes.

BookInfo Metainformation for a Book, in which it may appear. BookInfo must contain a BookBiblio, followed by any number of LegalNotices, followed by any number of ModeSpecs (which are pointed to by the LinkMode attribute of OLink, and are collected here for convenience). BookInfo has common attributes and a Contents attribute, the values of which are the IDs of the ToC, LoTs, Prefaces, Parts, Chapters, Appendixes, References, Glossary, Bibliography, and indexes comprising the Book, in the order of their appearance.

BridgeHead A free-floating heading not tied to the Sect hierarchy. It may contain in-line elements and has common and Renderas attributes. Use the Renderas attribute to indicate the format in which the BridgeHead should appear (Sect1, Sect2, Sect3, Sect4, Sect5, or Other; the default is IMPLIED).

Caution An admonition set off from the text; Tip, Warning, Important, and Note all share its model. Its contents may include paragraphs, lists, and so forth, but not another admonition. Caution and its sisters have common attributes.

Chapter A part of a Book. Chapter contain anything except higher-level elements such as Part, Book, and Set, or peers such as Appendix and Preface. A Chapter may begin with a DocInfo, followed by a required Title, optional TitleAbbrev, and body. The body may be either paragraphs and block-oriented elements or Sects containing them. At the beginning and end of the body of a Chapter, or the beginning and end of any Sect, there may be any number of ToCs, LoTs, Indexes, Glossaries, and Bibliographies, not that such an organization is recommended for ordinary use. Chapter has common and Label attributes; the Label attribute is for holding a numeric designation (3 for Chapter 3, for example).

Citation Any in-line bibliographic reference to another published work that uses a reference string, such as an abbreviation in a Bibliography. Compare CiteTitle. Citation contains plain text and has common attributes.

CiteTitle A citation for some published work. Compare Citation. It may contain in-line elements and plain text, and has common and Pubwork attributes. The value of Pubwork should indicate the type of work cited; it may be Article, Book, Chapter, Part, RefEntry, or Section (no default). CiteTitle is not a link or a cross reference, only a citation.

CiteRefEntry A citation of a reference entry. It must have a RefEntryTitle, followed by an optional ManVolNum. It has common attributes.

City Part of Address. It contains plain text and has common attributes.

Classname The name of the class to which a program component belongs. It contains plain text and has common attributes.

CmdSynopsis Synopsis for a Command. A CmdSynopsis contains any number of Args and Groups, in any order, followed by a single required Command, another set of any number of Args and Groups, in any order, and ending with any number of optional SynopFragments. CmdSynopsis has common, Label, and Sepchar attributes. Label has no default; Sepchar has the default " ". The following is courtesy Eve Maler (see also the entries for the subelements elsewhere in this list:

An argument at the command level (Arg) is a parameter passed to a command to specify or modify the command's behavior; it often consists of a variable value (Replaceable) such as a filename field, or an option (Option) and its own nested argument (Arg). If the contents of an argument are not marked up further, they are assumed to be something the user must type as shown (probably an option, but this is implicit). The Choice attribute on the two argument elements indicates whether the argument must be present ("Req", the default), whether it is optional ("Opt"), or whether no indication of presence is given ("Plain", the default).

A group (Group) as a peer or child of an argument is a collection of arguments, options, or other constructs that must appear in some relation to each other. For example, three options that are exclusive of each other and are optional would be in a group with a choice of "Opt" (the default). The "Optmult" choice allows zero or more of the children of the group to be supplied, and the "Reqmult" choice requires one or more of the children of the group to be supplied.

An option (Option) is a literal string or name of a parameter-keyword controlling the behavior of the command; a variable value (Replaceable) is a mnemonic name for a value that the user must supply, such as an input file name. The Option and Replaceable elements are available in text as well as in synopses. Stacks of options (for example, "-aefstuv") should be put into a single Option element for simplicity.

Depressingly complex constructs may appear anywhere within a synopsis. A SynopFragment reference (SynopFragmentRef) is a special kind of variable value assigned in place of this construct, which is then broken out into its own synopsis subset (SynopFragment) for clarity. A SynopFragment must have an ID, and any SynopFragmentRefs supplied must point to some SynopFragment.

Plain text within a CmdSynopsis is allowed only inside Cmd, Arg, Option, Replaceable, and SynopFragmentRef. The top-level separator character attribute value (" " by default) should be used to separate arguments and groups from their repeat indicators (". . .") and to separate Commands and their arguments and groups at the top level.

The CmdSynopsis structure does not meet the needs of DCL and other VMS-related command languages that have command parameters such as /[NO]WRITE, positional versus nonpositional parameters, and so on. Probably additional low-level elements would have to be added to the mix and the top-level structure enhanced slightly to account for these. However, CmdSynopsis appears to meet most UNIX-related needs.

Processing expectations: The "Opt" settings on arguments and groups (and probably "Optmult" as well for now) should produce square brackets. The "Req" settings (and probably "Reqmult" as well for now) should produce curly braces. The children of Group (if there is more than one child) should be either separated by vertical bars or formatted as a stacked list. Spacing at the Command and Group levels is controlled by formatter defaults and/or the sepchar setting.

```
Example command synopsis in typical UNIX(tm) format:

  rm [-f] [-r] [-i] [-] filename|dirname . . .
  |    |    |    |   |  |   |       |    |
  |    optional args    |   |       |    repeat indicator
  |    (contain options)|   |       |
  |                     |   |       second child of group
  command name          |   |
|   first child of group
|
required repeatable group

     SGML source for this example:
<CMDSYNOPSIS>
<COMMAND>rm</COMMAND>
<ARG Choice="opt">-f</ARG>        (<OPTION> not required for arg contents
<ARG Choice="opt">-r</ARG>        unless doing extra-special processing)
<ARG Choice="opt">-i</ARG>
<ARG Choice="opt">-</ARG>         (various synopsis formats
<GROUP Choice="req" Rep="repeat">      can be generated)
```

```
<REPLACEABLE>filename</REPLACEABLE>
<REPLACEABLE>dirname</REPLACEABLE>
</GROUP>
</CMDSYNOPSIS>
```

A collaborative group of authors. It contains a required CollabName followed by any number of Affiliations, and has common attributes.

CollabName Name of a collaborative group of authors. It contains plain text and has common attributes.

ColSpec Column specifier, that is, formatting information for a column in a Table, part of TGroup, THead, or TGroup. ColSpec is an empty element, bearing common and Align, Char, Charoff, Colname, Colnum, Colsep, Colwidth, and Rowsep attributes. The default values come from the TGroup, THead, or TFoot that starts the current enclosing group. Each ColSpec is for a single column, so it properly has a column number, Colnum, implicitly in order starting from 1, and an optional Colname by which it is known when used in any SpanSpec or in Entry. A ColSpec set on THead or TFoot should be complete for all columns. It overrides those on the containing TGroup and applies to just the THead or TFoot. If there is no ColSpec used within THead or TFoot, then the ColSpec of the containing TGroup (or the prior TGroup) is used. ColSpecs from the containing TGroup apply to TBody.

For *TGroupStyle*, see *TGroup*.

Align controls the horizontal position of text within the column. The value of Align may be Left (quad flush left), Center (centered), Right (quad flush right), Justify (both quad left and right), or Char (align to the left of Char, positioned by Charoff). There is no default.

Char contributes to Align. If the value of Align is "char", the value of Char should be a character on the first occurrance of which the entry is to be aligned. If that character does not occur in the entry, the entry is aligned to the left of the position determined by Charoff. (Think of using the decimal point to align decimal numbers: Char=". "and Charoff says how far over in the column the decimals should be; if no decimal occurs, the number is an integer and is aligned to the left of the decimals.) No value implies there is no character to align on. The default is implied, from the enclosing TGroup. (That is, you normally declare that an entire TGroup shall be decimal-aligned, but if you need to align a specific column differently, you can do it by specifying another Char at the ColSpec level.)

Charoff contributes with Char to Align. Charoff is the proportion of the current column width, expressed in percentage, to be allowed before the left edge of the first occurrence of the character given as the value of Char. The default is inherited from the enclosing TGroup. That is, if columns in a TGroup are to be decimal-aligned, and the decimal point is to fall three-quarters of the way across each cell (most of your numbers are of the form 123.4), you could set Charoff to 75 in TGroup; that value would be inherited by ColSpec, where you could modify it for a specific column (for example, by setting it to 50 for a column of numbers of the form 12.34).

Colname gives the name of the column, which is used to specify the position in a row, or the start or end of a horizontal span of columns (SpanSpec). There is no default.

Colnum gives the number of the column, counting from 1 at left of the table. There is no default.

Colsep determines column separators. If its value is 1 (yes), display the internal column rulings to the right of each item; if 0 (no), do not display it. It is ignored for the last column, where the Frame setting applies. There is no default. The value is inherited from TGroupStyle if used.

Colwidth is either a proportional measure of the form *number**, such as "5*" for 5 times the proportion, or "*" (="1*"); or a fixed measure, such as 2pt for 2 points, 3pc for 3 picas; or a mixed measure, such as 2*+3pt. Coefficients are positive numbers with up to two decimal places. There is no default. If no value is given, the value should be obtained from the FOSI, or, if there is no FOSI value, the value 1 should be used. (Perhaps this means that you can vary the width of columns by stating their relative proportions, or you can give fixed widths. If you use a decimal number less than one, express it in the form 0.2, not .2.)

Rowsep determines row separators. If its content is 1 (yes), display the internal vertical row ruling below each item; if 0 (no), do not display it. It is ignored for the last row of the table, where the value of Frame applies to the entire Table. There is no default. The value is inherited from TGroupStyle, if used.

Command An executable program, or the entry a user makes to execute a command. It may contain members of cptrphrase.gp, and has common and MoreInfo attributes. For the MoreInfo attribute see *Application.*

Comment A remark made within the document file that is intended for use during interim stages of production. A Comment should not be displayed to the reader of the finished, published work. It may appear almost anywhere, and may contain almost anything below the Section level. Note that, unlike an SGML comment, unless you take steps to suppress it, the Comment element will be output by an SGML parser or application. You may wish to do this to display Comments along with text during the editorial process.

ComputerOutput Data presented to the user by a computer. It may contain elements from cptrphrase.gp, and has common and MoreInfo attributes For the MoreInfo attribute see *Application.*

ConfDates Dates of a conference in connection with which a document was written. It contains plain text (e.g., 21–24 May 1927) and has common attributes.

ConfGroup A wrapper for information about a conference. It contains any number of ConfDates, ConfTitles, ConfNums, Addresses, and ConfSponsors, in any order, and has common attributes.

ConfNum The number of a conference. It contains plain text, and has common attributes.

ConfSponsor Sponsor of a conference in connection with which a document was written. It contains plain text and has common attributes.

ConfTitle Title of a conference in connection with which a document was written. It contains plain text and has common attributes.

ContractNum Number of a contract under which a document was written. It contains plain text and has common attributes.

ContractSponsor Sponsor of a contract under which a document was written. It contains plain text and has common attributes.

Contrib Holds information about the contributions of an Author, Editor, or OtherCredit to the work in question; it may occur within those elements and contains plain text. It has common attributes.

Copyright Copyright information about a document. It consists of one or more Years followed by any number of Holders. It has common attributes.

CorpAuthor Corporate author of a book, for use in BookInfo or BiblioEntry. It contains plain text and has common attributes.

CorpName Name of a corporation. It contains plain text and has common attributes.

Country Part of Address. It contains plain text and has common attributes.

Data Optional component of FuncParam and ParamDef. It wraps plain text, Replaceable, more Data, Emphasis, and may contain links and indexing information. It has common attributes.

Date Date of publication or revision. It contains plain text. (No provision has been made for representing eras; you could include this information along with the date data.) It has common attributes.

Database An organized set of data. It may contain members of cptrphrase.gp, and has common, Class, and MoreInfo attributes. Class may have the value Name, Table, Field, Key1, Key2, or Record (no default). For the MoreInfo attribute see *Application.*

DocInfo Metainformation for a book component, in which it may appear. Only Title and AuthorGroup are required. DocInfo may contain, in order: the required Title, optional TitleAbbrev and Subtitle, followed by one or more AuthorGroups, any number of Abstracts, an optional RevHistory, and any number of LegalNotices. DocInfo has common attributes.

Edition The edition of a document. It contains plain text. It has common attributes.

Editor The editor of a document. Contents are the same as for *Author.*

Email Part of Address. It contains plain text and has common attributes.

Emphasis Provided for use where you would traditionally use italics or bold type to emphasize a word or phrase. It contains plain text and has common attributes.

Entry A euphemism for a cell in a table—you'd rather be in an Entry than in a Cell, wouldn't you? An Entry occurs in a Row, and must have either some assortment of paragraphs, block-oriented elements, and Graphics, or in-line elements. Entry has common and Align, Char, Charoff, Colname, Colsep, Morerows, Nameend, Namest, Rotate, Rowsep, Spanname, and VAlign attributes. Attribute values may be inherited from enclosing elements that share the same attribute. "An Entry not specified by a SpanSpec gets its defaults from its starting column."

Align controls the horizontal position of text within the column. The value of Align may be Left (quad flush left), Center (centered), Right (quad flush right), Justify (both quad left and right), or Char (align on leftmost of Char, positioned by Charoff). There is no default; the value may be inherited from ColSpec or SpanSpec.

Char contributes to Align. If the value of Align is "char", the value of Char should be a character on the first occurrance of which the entry is to be aligned. If that character does not occur in the entry, the entry is aligned to the left (the original doc incorrectly specifies "right") of the position determined by Charoff. The default is inherited; no value implies there is no character to align on.

Charoff contributes with Char to Align. Charoff is the proportion of the current column width, expressed in percentage, to be allowed before the left edge of the first occurrence of the character given as the value of Char, if any. The default is inherited from ColSpec or SpanSpec.

Colname gives the name of the column, which is used to specify the position in a row, or the start or end of a horizontal span of columns (SpanSpec). There is no default; omit if SpanName is present. The implied value is either the first column of the Row, or, if already in a Row, the next column after the end of the prior Entry or EntryTbl.

Colsep determines column separators. If its value is 1 (yes), display the internal column rulings to the right of each Entry; if 0 (no), do not display it. It is ignored for the last column, where the Frame setting applies. (In CALS, Yes is expressed as 1 and No as 0.) There is no default; if no value is given the value is inherited from ColSpec or SpanSpec.

Morerows is the number of additional rows in a vertical straddle. The default is 0.

Nameend is the name of the rightmost column of a span. Names are identified in the ColSpec of the current TGroup. There is no default.

Namest or Name Start, is the name of the leftmost column of a span. Names are identified in the ColSpec of the current TGroup.

Rotate governs rotations, which are not additive to those specified in the FOSI. Values may be 1 (yes) or 0 (no). 0 (no) specifies no rotation; 1 (yes) specifies 90 degrees rotation counterclockwise to table orientation. No other values are supported!

Rowsep determines row separators. If its content is 1 (yes), display the internal vertical row ruling below each item; if 0 (no), do not display it. It is ignored for the last row of the table, where the frame value applies. There is no default. The value is inherited from Row, if used there.

Spanname is the name of a horizontal span. No default.

VAlign governs the vertical positioning of text within an Entry. Allowed values are Top, Middle, and Bottom (no default).

EntryTbl A form of subtable. It may occur in Row, along with Entry. Several EntryTbls of differing formats may occur in the same Row of a TBody, but EntryTbl may not contain itself. Aside from that restriction, EntryTbl contains one or more sets of these elements, in order: any number of ColSpecs, any number of SpanSpecs, an optional THead, and a required TBody. There is no implication of alignment of subrows in different EntryTbls. Default attribute values come instead from those of like-named attributes on enclosing elements: Table, TGroup, ColSpec, SpanSpec, THead, TFoot, TBody, or Row. EntryTbl has common, Align, Char, Charoff, ColName, Cols, Colsep, Nameend, Namest, Rowsep, Spanname, and TGroupStyle attributes.

Align controls the horizontal position of text within the column. The value of Align may be Left (quad flush left), Center (centered), Right (quad flush right), Justify (both quad left and right), or Char (align to the left of Char, positioned by Charoff). There is no default; the value may be inherited from ColSpec or SpanSpec.

Char contributes to Align. If the value of Align is "char", the value of Char should be a character on the first occurrance of which the entry is to be aligned. If that character does not occur in the entry, the entry is aligned to the left (the original doc incorrectly specifies "right") of the position determined by Charoff. There is no default.

Charoff contributes with Char to Align. Charoff is the proportion of the current column width, expressed in percentage, to be allowed before the left edge of the first occurrence of the character given as the value of Char, if any. The default is inherited from the enclosing TGroup.

Colname gives the name of the leftmost column of EntryTbl. There is no default.

Cols is the number of columns in the EntryTbl. There is no default.

Colsep determines column separators. If its value is 1 (yes), display the internal column rulings to the right of the EntryTbl, except if the EntryTbl falls in the the last column, where the siderule (*sic*) setting applies; if 0 (no), do not display it. There is no default. The value is inherited from the enclosing TGroup.

Nameend is the name of the rightmost column of a span. Names are identified in the ColSpec of the current TGroup. There is no default.

Namest or Name Start, is the name of the leftmost column of a span. Names are identified in the ColSpec of the current TGroup. There is no default.

Rowsep determines row separators. If its content is 1 (yes), display the internal vertical row ruling below the EntryTbl; if 0 (no), do not display it. It is ignored for the last row of the table, where the frame value applies. There is no default. The value is inherited from the enclosing TGroup.

Spanname is the name of a horizontal span.

TGroupStyle is the name of a table group style defined in the FOSI. There is no default.

Epigraph A brief section of poetry or prose at the start of a chapter. It contains paragraphs and has common attributes.

Equation A titled mathematical equation displayed on a line by itself, rather than in-line. It has an optional Title and TitleAbbrev, followed by either an InformalEquation or a Graphic (see *Graphic*). Equation has common and Label attributes.

ErrorName An error message reported by a computer. It contains plain text and has common attributes.

ErrorType A classification of an error message reported by a computer. It contains plain text and has common attributes.

Example Intended for sections of program source code that are provided as examples in the text. It contains a required Title and an optional TitleAbbrev, followed by one or more block-oriented elements in any combination. It has common and Label attributes. A simple Example might contain a Title and a ProgramListing.

Fax Part of Address. It contains plain text and has common attributes.

Figure An illustration. It must have a Title, and may have a TitleAbbrev, followed by one or more of BlockQuote, InformalEquation, Graphic, InformalTable, Link, LiteralLayout, OLink, ProgramListing, Screen, Synopsis, and ULink, in any order. Figure has common, Label, and Float attributes; Float indicates whether the Figure is supposed to be rendered where convenient (yes) or at the place it occurs in the text (no, the default). To reference an external file containing graphical content use the Graphic element within Figure.

Filename The name of a file, including pathname if this information is present. It may contain members of cptrphrase.gp, and has common and MoreInfo attributes. For the MoreInfo attribute see *Application.*

FirstName (Western-style) given name of Author, Editor, or OtherCredit. It contains plain text. It has common attributes.

FirstTerm First occurrence of a word in a given context. It contains plain text and has common attributes.

Footnote The contents of a footnote. The point in the text where the mark for a specific footnote goes is indicated either by FootnoteRef or by the actual placement of the Footnote itself (in which case FootnoteRef may be omitted). Footnote may contain Para, SimPara, BlockQuote, InformalEquation, InformalTable, Graphic, Synopsis, LiteralLayout, ProgramListing, Screen, and any kind of list. It has optional ID, Label, Lang, Remap, Role, and XRefLabel attributes.

FootnoteRef Identifies the location for a footnote mark. It may contain plain text, which is the mark to be displayed, or it may be empty, in which case you can use the Mark attribute to indicate the contents of the mark (such as an asterisk, *, a number, 84, or a dingbat specified by a name that is to be interpreted by the application). FootnoteRef has ID, Linkend, and Mark attributes. The Linkend attribute, which is required, has as its value the ID of the associated Footnote. See *Footnote.*

ForeignPhrase Any word or words from a language other than that of the document which you want to mark off in some way. In English, *inter alia* and *c'est la vie* are ForeignPhrases. It contains plain text and has common attributes.

FormalPara A paragraph with a Title. FormalPara contains a required Title followed by a required Para, and has common attributes.

FuncDef Part of a FuncSynopsis. Like Paramdef, it provides data type information and the name of the Function (or Parameter, in the case of ParamDef) this information applies to. A FuncDef may contain any combination of plain text, Replaceable, Data, or Function, in any order. It has common attributes.

Separating this information from the rest of the synopsis avoids messing with data type information that appears before *or after* the item it applies to, such as array information ("[]"). It also avoids the issue of placing the pointer ("*") indicator (next to the rest of the left-hand data type or next to the Parameter or Function name?). Any spaces that surround the Parameter or Function must be inserted by the writer.

FuncParams Optional component of ParamDef. It supplies "inner parameters" for Paramters that are pointers to Functions. FuncParams contains elements from ctprphrase.gp and has common attributes.

FuncSynopsis (contributed Eve Maler, along with remarks on its subelements.) A C synopsis that shows a prototype or definition indicating a function's name. A FuncSynopsis also indicates the data type of its return value, and the positions, purposes, and data types of its parameters. A FuncSynopsis begins with an optional FuncSynopsisInfo, which contains additional information about the synopsis that follows; line breaks and leading white space are significant within a FuncSynopsisInfo. This is followed by one or more blocks defining a function (you might use more than one for connecting related sets of functions). Each of these blocks consists of a required FuncDef, followed by a Void, a VarArgs, or one or more ParamDefs. Void, Varargs, and and ParamDef are mutually exclusive. *Usage Note:* You should supply no specific information on the arguments before the ellipsis that VarArgs should output when rendered, but if it is necessary to represent the ellipsis in the source it may ben enclosed within a final ParamDef. FuncSynopsis has common and Label attributes.

The processing application is expected to provide all parentheses, semicolons, and the like. The exceptions are any spaces surrounding function and parameter names, any parentheses or commas or spacing inside lists of data types of parameters that are pointers to functions, and the parentheses around those parameter names themselves. These exceptions are a bit confusing in the unusual case (pointers to functions), but they *greatly* simplify tagging and still allow either K&R style or ANSI C style to be produced (assuming writers have cooperated in supplying enough information for ANSI).

FuncSynopsisInfo Information supplementing the FuncDefs of a FuncSynopsis. It contains elements of ctprphrase.gp, and within it line breaks and leading white space are significant. See *FuncSynopsis.* It has common attributes.

Function A subroutine in a program or external library. It may contain members of cptrphrase.gp, and has common and MoreInfo attributes. For the MoreInfo attribute see *Application.*

Glossary A glossary of terms. Glossary may occur within a Chapter, Appendix, or Preface, or may be a book component in its own right. It contains in order an optional DocInfo, optional Title, and optional TitleAbbrev, followed by any number of block-oriented elements, followed by one or more GlossEntries or one or more GlossDivs, and ending with an optional Bibliography. It has common attributes.

GlossDef The definition attached to a GlossTerm in a GlossEntry. It may contain Comments, GlossSeeAlsos, paragraphs, and other block-oriented elements, in any order; it has common and Subject attributes. The Subject attribute may hold a list of subject areas (e.g., DCE RPC General) as keywords.

GlossDiv A division of a Glossary. It may have a Title and TitleAbbrev, followed by block-oriented elements, followed by one or more GlossEntries. It has common attributes.

GlossEntry An entry in a Glossary. It contains, in order, a required GlossTerm, an optional Acronym, an optional Abbrev, and any number of GlossSees and GlossDefs, in any order. It has common and Sortas attributes, the latter to hold the string by which the GlossTerm is to be sorted (alphabetized) in lieu of its proper content.

GlossList GlossList is a wrapper for a set of GlossEntries. It has common attributes.

GlossSee A cross-reference from one GlossEntry to another. It may contain plain text and inline elements, or no content, and has common and OtherTerm attributes. OtherTerm is a reference to the GlossTerm within the cross-referenced GlossEntry; that GlossTerm should be displayed at the point of the GlossSee.

GlossSeeAlso A cross-reference from one GlossDef to another GlossEntry. It may contain plain text and in-line elements, or no content, and has common and OtherTerm attributes. OtherTerm is a reference to the GlossTerm within the cross-referenced GlossEntry; that GlossTerm should be displayed at the point of the GlossSee.

GlossTerm A term in the text of a Chapter (for example) that is glossed in a Glossary; also used for those terms in GlossEntries, in the Glossary itself. As you may not want to tag all occurrences of these words outside of Glossaries, you might consider GlossTerm, when used outside of Glossaries, to be similar to FirstTerm, except that GlossTerm may contain other in-line elements. GlossTerm contains in-line elements and has common attributes.

Graphic Encloses graphical data or points via an attribute to an external file containing such data, and is to be rendered as an object, not in-line. It has Format, Fileref, Entityref, SrcCredit, and ID attributes. The format attribute may have the value of any of the formats defined at the head of the DTD: CGM-BINARY, CGM-CHAR, CGM-CLEAR, DITROFF, DVI, EPS, EQN, FAX, GIF, IGES, PIC, PS, TBL, TEX, TIFF.

The value of Fileref should be a filename, qualified by a pathname if desired; the value of Entityref should be that of an external data entity. If data is given as the content of Graphic, both Entityref and Fileref, if present at all, should be ignored, but a Format value should be supplied. if no data is given as the content of Graphic and a value for Entityref is given, Fileref, if present, should be ignored but no Format value should be supplied. Finally, if there is no content for Graphic and Entityref is absent or null, Fileref must be given the appropriate value, and again no Format value should be supplied.

Group A group of constituent parts of a CmdSynopsis. A Group consists of one or more Args, Groups, SynopFragmentrefs, and Replaceables, in any order. See *CmdSynopsis.* Group has common, grpchcatt, and repatt attributes.

argchcatt or "argument choice attribute" resolves to the attribute Choice, with allowed values Opt (the Arg is optional; the default), Req (it is required), and Plain (neither optional nor required).

repatt or "repetition attribute" resolves to the attribute Rep, with allowed values Norepeat (the Arg may be repeated; the default) and Repeat (the Arg does not repeat).

grpchcatt or "group choice attribute" is a parameter entity that resolves to the the Choice attribute for the element Group. The allowed values are Opt (the Arg is optional; the default), Req (it is required), Plain (neither optional nor required), OptMult (optional and repeatable), and ReqMult (required multiple times).

Hardware A physical part of a computer system. It may contain members of cptrphrase.gp, and has common and MoreInfo attributes. For the MoreInfo attribute see *Application.*

Highlights A list of main points discussed in a book component such as a Chapter. It may contain paragraphs, lists, and admonitions, and has common attributes.

Holder Part of Copyright; the holder of the copyright of the document. It contains plain text. It has common attributes.

Honorific A person's title, to be used as part of Author, Editor, or OtherCredit. It contains plain text. It has common attributes.

Important An admonition set off from the text. See *Caution.*

Index An index to a Chapter, Appendix, Preface, or Book. It contains an optional DocInfo, Title, and TitleAbbrev, followed by any number of block-oriented elements, and then one or more IndexEntries or one or more IndexDivs. It has common attributes.

IndexDiv A division of an Index. It may have a Title and TitleAbbrev, some optional introductory matter (block-oriented elements, Anchors, Comments), and must then contain one or more IndexEntries or a SegmentedList (use a SegmentedList for a permuted index). It has common attributes.

IndexEntry Part of Index. It contains a PrimaryIE, which may be followed by any number of SeeIEs and SeeAlsoIEs; these may be followed by any number of sets of SecondaryIEs followed by TertiaryIEs, SeeIEs, and SeeAlsoIEs. It has common attributes.

IndexTerm A character string to be indexed, occurring in the text flow but not in the text itself. (And remember, IndexTerm appears in the text, not in the Index!) An IndexTerm must begin with a Primary, which may be followed by a Secondary, a See, or any number of SeeAlsos. A Secondary, in turn, may be followed by a Tertiary, a See, or any number of SeeAlsos; and a Tertiary may also be followed by a See or any number. of SeeAlsos IndexTerm has common, SpanEnd, PageNum, Significance, Scope, and Zone attributes. The SpanEnd attribute may not be used if IndexTerm has content; it should be used only to mark the end of a span of text that begins earlier at an IndexTerm that does have content. The value of SpanEnd must be the ID of that earlier IndexTerm. The PageNum attribute may be used to indicate the page on which the indexed term is found in print. Significance may have the value Preferred, indicating that the entry is the most pertinent of the series, or Normal (the default). The Scope attribute allows you to indicate what generated indices the IndexTerm should appear in: All, Global, or Local (all the indices, only the indices for collections, such as Sets, or only the index for the work at hand). The default value is IMPLIED, so you should set your implementation's default to suit your case. The Zone attribute takes IDREFs, which should be the IDs of elements to which the IndexTerm applies. This attribute allows you to attach indexing or keyword information to objects rather than just to points in the text.

InformalEquation An untitled mathematical equation displayed on a line by itself, rather than in-line. It contains a Graphic, and has common attributes.

InformalExample An untitled Example. See *Example.* It has common attributes.

InformalTable An array of text that has no Title. Otherwise, it is just like Table except that it lacks the ShortEntry and ToCEntry attributes. See *Table.*

InlineEquation An untitled mathematical equation occurring in-line or as the content of an Equation. It contains a Graphic, and has common attributes.

InlineGraphic Encloses graphical data or points via an attribute to an external file containing such data, and is to be rendered in-line. InlineGraphic has Format, Fileref, Entityref, SrcCredit, and ID attributes. The format attribute may have the value of any of the formats defined at the head of the DTD, under "Notations." If it is desired to point to an external file, a filename may be supplied as the value of the Fileref attribute, or an external entity name may be supplied as the value of the Entityref attribute. SrcCredit can be used to hold information about the source of the image. Unlike Graphic, InlineGraphic should cause no line break that would not occur for proper line justification.

Interface Any part of a graphical user interface. It may contain members of cptrphrase.gp, and has common, Class, and MoreInfo attributes. Class may have the value Button, Icon, Menu, or MenuItem (no default). For the MoreInfo attribute see *Application.*

InterfaceDefinition A specification for a graphical user interface. It may contain members of cptrphrase.gp, and has common and MoreInfo attributes. For the MoreInfo attribute see *Application.*

InvPartNumber An inventory part number. It contains plain text and has common attributes.

ISBN International Standard Book Number of a document. It contains plain text and has common attributes.

ISSN International Standard Serial Number of a journal. It contains plain text and has common attributes.

IssueNum The number of an issue of a journal. It contains plain text and has common attributes.

ItemizedList A list in which each item is marked with a bullet, dash, or other dingbat (or no mark at all). It consists of one or more ListItems. A ListItem in an ItemizedList contains paragraphs and other block-oriented elements, which may in turn contain other lists; an ItemizedList may be nested within other lists, too. It has common attributes and a Mark attribute. Your application might supply the mark to be used for an ItemizedList, but you can use this attribute to indicate the mark you desire to be used; there is no fixed list of these. *Usage Note:* You might want to use one of the ISO text entities that designates an appropriate dingbat.

JobTitle Part of Affiliation. It contains plain text and has common attributes.

JournalInfo Information about the journal in which an Article appears, in that Article's ArtHeader. It contains, in order, a required Title, optional TitleAbbrev, Subtitle, ISSN, VolumeNum, IssueNum, PageNums, PubDate, Publisher, and Copyright.

KeyCap The text printed on a physical key on a computer keyboard, not necessarily the same thing as a KeyCode. It may contain members of cptrphrase.gp, and has common and MoreInfo attributes. For the MoreInfo attribute see *Application.*

Keycode The computer's numeric designation of a key on a computer keyboard. Keycode contains plain text and has common attributes.

Keysym A key symbol name, which is not necessarily the same thing as a Keycap. For example, the Keysym for the H key (Keycap H) might be *h.* It contains plain text, and has common attributes.

LegalNotice An acknowledgement of trademarks, etc. It may have a Title, followed by paragraphs and most kinds of block-oriented elements, in any order. It has common attributes.

Lineage Part of an author's name, such as "Jr." It contains plain text and has common attributes.

LineAnnotation A writer's or editor's comment on a line of program code within an Example, ProgramListing, or Screen. LineAnnotations are a document author's comments on the code, not the comments written into the code itself by the code's author.

Link A hypertext link. At present, all the link types represented in the DTD are provisional. Link is less provisional than the others, however. In Hytime parlance, Link is a clink. It may contain in-line elements and has Endterm, Linkend, and Type attributes. The required Linkend attribute specifies the target of the link, and the optional Endterm attribute specifies text that is to be fetched from elsewhere in the document to appear in the Link. You can also supply this text directly as the content of the Link, *in which case the Endterm attribute is to be ignored.* You may assign any value to the Type attribute.

ListItem A wrapper for the elements of items in an ItemizedList or OrderedList; it also occurs within VarListEntry in VariableList. It may contain just about anything except Sects and book components. It has common attributes and an Override attribute, which may have any of the values of ItemizedList's Mark attribute; use Override to override the mark set at the ItemizedList level, when you desire to create ItemizedLists with varying marks.

Literal Any literal string, used in-line, that is part of data in a computer. This may be as precise as the value of an argument, but Literal may also be used as a catch-all element. It may contain members of cptrphrase.gp, and has common and MoreInfo attributes. For the MoreInfo attribute see *Application.*

LiteralLayout The wrapper for lines set off from the main text that are not tagged as Screens, Examples, or ProgramListing, in which line breaks and leading white space are to be regarded as significant. It contains in-line elements, and has common and Width attributes, for specifying a number representing the maximum width of the contents.

LoT The generic tag for such things as a List of Figures or List of Tables. An LoT may occur within a Chapter, or Appendix, or may be a book component on its own. It contains, in order, an optional DocInfo, Title, and TitleAbbrev, followed by one or more LoTentries. It has common, SrcCredit, and Label attributes; in this case the values of Label may be Equation, Examples, Figures, or Tables, with no default.

LoTentry An element of LoT. It contains the text of the thing to be listed, including, if desired, in-line elements. It has common, SrcCredit, and PageNum attributes. SrcCredit can be used to hold information about the source of the entry in the case of a list of illustrations, and PageNum should be used to indicate either the page on which the entry begins or the pages it covers.

ManVolNum Specific to UNIX man pages, it designates the section of a complete set of reference pages that a reference page belongs to. It appears within RefMeta, contains plain text, and has common attributes.

Markup A string of formatting markup in text, which it is desired to represent literally. See also *SGMLTag.* It contains plain text and has common attributes.

MediaLabel The physical medium on or in which some information is contained. MediaLabel may contain plain text, and has common, Class, and MoreInfo attributes. Class may have the value Cartridge, CDRom, Disk, or Tape (no default). For the MoreInfo attribute see *Application.*

Member Part of a SimpleList. It contains in-line elements and has common attributes.

ModeSpec Contains application-specific information necessary for the completion of an OLink; see *OLink.*

Msg The part of a MsgEntry that contains the error message and its subparts, along with explanatory text. A Msg has a required MsgMain, followed by any number of MsgSubs and MsgRels, in any order. It has common and Label attributes.

MsgAud Describes the audience to which a Msg is relevant. It contains plain text only, and has common attributes.

MsgEntry A wrapper for an entry in a MsgSet. A MsgEntry must contain one or more Msgs, followed by an optional MsgInfo, then any number of MsgExplans. MsgEntry has common attributes.

MsgExplan Holder for any kind of explanatory material relating to the Msg. MsgExplan begins with an optional Title (typically something such as “Explanation:” or “Action:”) and may contain block-oriented elements. It has common attributes.

MsgInfo Information about the Msg containing it. It may have any number of MsgLevels, MsgOrigs, and MsgAuds, in any order, and has common attributes.

MsgLevel The level of importance or severity of a Msg. It contains only plain text, and has common attributes.

MsgMain The main error message of a Msg. MsgMain begins with an optional Title and contains MsgText, which is the text of the message. It has common attributes.

MsgOrig The origin of a Msg. It contains only plain text and has common attributes.

MsgRel An optional subpart of a Msg, containing a message that is related to the main message (MsgMain) but which appears in a different place. For example, MsgMain might be a message that appears on a network client, and MsgRel a related message that appears at the server console in response to the same condition or event. MsgRel begins with an optional Title and contains MsgText. It has common attributes.

MsgSet A list of error messages produced by a system, with various additional information. MsgSet contains one or more MsgEntries, and has common attributes.

MsgSub An optional subpart of a Msg, which might contain messages that appear in various contexts. It contains a an optional Title followed by MsgText. MsgSub has common attributes.

MsgText Contents of the parts of Msg. It may contain block-oriented elements, and has common attributes.

Note A message to the user, set off from the text. See *Caution.*

OLink A link that may perform some operation to find its target. In contrast to Link, OLink has no Linkend attribute, but rather a TargetDocEnt, the value of which is the name of a text or data entity already defined by the user. The LinkMode attribute points by ID to a ModeSpec; for convenience, ModeSpecs located in the BookInfo. ModeSpec contains instructions (probably application-specific) for operating on the entity named by TargetDocEnt, e.g., the TargetDocEnt is another Book, and the ModeSpec specifies that all the second-level headings should be searched for a phrase. The LocalInfo attribute may be used to hold such a phrase, which may be thought of as a replacement for some variable in ModeSpec.

OLink may contain in-line elements and has a Type attribute, to which you may assign any value.

Option An option for a computer program command. It may have members of cptrphrase.gp, and has common attributes.

Optional For use in Synopsis, as in a RefEntry, where optional parameters conventionally are shown in square brackets; may also appear in-line. Optional should replace those brackets. It may contain elements from cptrphrase.gp, and has common attributes.

OrderedList A numbered or lettered list, consisting of ListItems. A ListItem in an OrderedList contains paragraphs and other block-oriented elements, which may in turn contain other lists; an OrderedList may be nested within other lists, too. OrderedList has common attributes, along with a Numeration attribute, which may have the value Arabic, Upperalpha, Loweralpha, Upperroman, or Lowerroman. If no value is supplied, the processing expectation should be that Arabic numbering (1, 2, 3, . . .) is to be used. It has an InheritNum attribute, for which the value Inherit specifies for a nested list that the numbering of ListItems should include the number of the item within which they are nested (2a, 2b, etc., rather than a, b, etc.); the default value is Ignore. It has a Continuation attribute, with values Continues or Restarts (the default), which may be used to indicate whether the numbering of a list begins afresh (default) or continues that of the immediately preceding list (Continues). You need supply the Continuation attribute only if your list continues the numbering of the preceding list.

OrgDiv Part of an organization. It contains plain text and has common attributes.

OrgName An organization that is not a corporation (*cf.* CorpName) It contains plain text and has common attributes.

OtherAddr (Formerly Other.) An escape hatch for information given in Address that doesn't fit in any other element. It has common attributes.

OtherCredit Supplements Author and Editor; you could use it to credit a contributor, a contributing editor, or some deserving production person. Contents are as for *Author.* It has common attributes.

OtherName An alternative to Firstname and Surname, for use in Author, Editor, or OtherCredit. It contains plain text and has common attributes.

PageNums The numbers of the pages contained in a Book, for use in its BookBiblio. It contains plain text (e.g., ix, 292) and has common attributes.

Para A paragraph. A Para may not have a Title: to attach a Title to a Para use FormalPara. Para may contain any in-line element and almost any block-oriented element. Abstract, AuthorBlurb, Caution, Important, Note, and Warning are excluded, as are Sects and higher-level elements. Para has common attributes. See also *FormalPara* and *SimPara.*

Paragraphs See *Para.*

ParamDef Part of a FuncSynopsis, providing data type information and the name of the Parameter this information applies to. A ParamDef may contain any combination of plain text, Replaceable, Data, or Parameter, in any order. It has common attributes. See *FuncDef, FuncSynopsis.* Parameter has common and Class attributes, the latter with the possible values Command, Function, and Option (no default).

Parameter Part of an instruction to a computer. It may contain members of cptrphrase.gp, and has common, Class, and MoreInfo attributes. Class may have the value Command, Function, or Option (no default). For the MoreInfo attribute see *Application.*

Part A section of a Book containing book components. Part contains an optional DocInfo, a required Title, an optional TitleAbbrev, an optional PartIntro, followed by one or more book components. It has common and Label attributes.

PartIntro Introduction to the contents of a Part; may also appear in Reference. It has optional Title and TitleAbbrev, and then may contain anything that can appear in a Chapter. PartIntro has common and Label attributes.

Phone Part of Address. It contains plain text and has common attributes.

POB Part of Address. It contains plain text and has common attributes.

Postcode Part of Address. It contains plain text and has common attributes.

Preface Any introductory matter in a Book. Preface may occur more than once in Book, before any Chapter, Part, or Reference: for example, you might have two Prefaces, one titled "Preface" and the other "Introduction." Preface may begin with a DocInfo, followed by a required Title, optional TitleAbbrev, and anything found in the body of a Chapter. Preface has common attributes.

Primary A word or phrase occurring in the text that is to appear in the index under as a primary entry. It must be nested within IndexTerm tags. Primary may contain in-line elements. It has SortAs and common attributes. SortAs can be used to provide an alternate string for alphabetizing the index: if Primary has the content "14" one might give Primary the attribute Sortas="fourteen".

PrimaryIE A primary entry in an Index, not in the text. It may contain plain text and inline elements. It has common attributes and an optional Linkends attribute, which may have the value of some list of element IDs.

PrintHistory The printing history of a Book. It contains paragraphs and has common attributes.

Procedure A list of operations to be performed. Procedure may have a Title and TitleAbbrev, followed by block-oriented elements, such as paragraphs, followed by one or more Steps. A Step may have a SubSteps wrapper for Steps nested within it, and this nesting may continue indefinitely (contrast the methods of nesting lists and Sects). This construction is intended to maximize the reusability of subsections of Procedures. Procedure has common attributes.

ProductName Formal name for any product. It contains in-line elements, and has common and Class attributes. The Class attribute may have the values Service, Trade (the default), Registered, or Copyright. Thus a trademark can be marked up as Trademark or as a Productname with the default Class attribute value.

ProductNumber A number assigned to a product. It contains plain text and has common attributes.

ProgramListing A listing of a program. Line breaks and leading white space are significant in a ProgramListing, which may contain in-line elements, including LineAnnotations. (LineAnnotations are a document author's comments on the code, not the comments written into the code itself by the code's author.) ProgramListing has common and Width attributes, the latter for specifying a number representing the maximum width of the contents.

Property A defined set of data associated with a window. It may contain members of cptrphrase.gp, and has common and MoreInfo attributes. For the MoreInfo attribute see *Application.*

PubDate The date of publication of a document. It contains plain text and has common attributes.

Publisher The publisher of a document. It contains a PublisherName and any number of Addresses. It has common attributes.

PublisherName The name of a publisher of a document. It contains plain text and has common attributes.

PubsNumber A number assigned to a publication, other than an ISBN or ISSN or InvPartNumber. It contains plain text and has common attributes.

Quote An in-line quotation. For block quotes use BlockQuote. Quote may contain plain text and in-line elements, and has common attributes.

RefClass An element of RefNameDiv, in which the applicability or scope of the topic of a RefEntry may be indicated. It may contain plain text or Application, and has common attributes.

RefDescriptor A substitute for RefName to be used when a RefEntry covers more than one topic and none of the topic names is to be used as the sort name. It contains plain text and has common attributes.

RefEntry A reference page. It contains, in order, an optional DocInfo and optional RefMeta; any number of Comments and members of links.gp, in any order; a required RefNameDiv, an optional RefSynopsisDiv, and one or more RefSect1s. It has common attributes.

Reference A collection of RefEntries, formin a book component. Reference has an optional DocInfo, a required Title, an optional TitleAbbrev, an optional PartIntro, and one or more RefEntries. It has common and Label attributes.

RefEntryTitle Primary name given to a reference page for sorting and indexing. It may be the same as the first of the RefNames, or it may be the same as the RefDescriptor. It may contain in-line elements, and has common attributes.

RefMeta The first major division of a reference page, in which metainformation about the reference page is supplied. RefMeta contains, in order, a required RefEntryTitle, an optional ManVolNum, and any number of RefMiscInfos. It has common attributes.

RefMiscInfo Marks information in RefMeta that may be supplied by vendors, such as copyright, release date, revision date, print status, operating system, hardware architecture, or a descriptive phrase for use in a print header. It contains plain text, and has common and Class attributes. The Class attribute can be used to distinguish categories of RefMiscInfos; it has no defined values.

RefName The subject or subjects of a reference page. It appears within RefNameDiv. It may contain plain text and in-line elements, and has common attributes.

RefNameDiv The second major division of a reference page. It contains, in order, an optional RefDescriptor, one or more RefNames, a required RefPurpose, and any number of optional RefClass, followed by any number of Comments and links, in any order. It has common attributes.

RefPurpose A short phrase describing the subject of the reference page. It may contain in-line elements and has common attributes.

RefSect1 Equivalent to a Sect1 in the DocBook DTD. It contains a Title, followed by any of the allowable contents of a Sect, except that only two levels of subsection are allowed, RefSect2 and RefSect3. It has common attributes.

RefSect2 Equivalent to a Sect2 in the DocBook DTD, and may occur within RefSect1 or RefSynopsisDiv. It may contain any of the allowable contents of a Sect, except that only RefSect3 is allowed as a further subsections. It has common attributes.

RefSect3 Subdivision of RefSect2. No further subdivisions allowed; contents otherwise as for RefSect2. It has common attributes.

RefSynopsisDiv The third major division of a reference page, in which the syntax of the subject of the reference page is indicated. It contains, in order, an optional Title and TitleAbbrev, followed by either one or more synopses (Synopsis, CmdSynopsis, or FuncSynopsis) succeeded by any number of RefSect2s, or simply one or more RefSect2s. It has common attributes.

ReleaseInfo Information about a particular version of a document. It contains plain text and has common attributes.

Replaceable Part of a synopsis or command line, indicating that its contents may be replaced. It may contain basic in-line elements (not cptrphrase.gp) and has common and Class attributes. Class may have the value Command, Function, Option, or Parameter (no default).

ReturnValue A value returned by a function. It contains plain text and has common attributes.

RevHistory A section of BookInfo or DocInfo recording revisions to the document. It consists of any number of Revisions, and has common attributes.

Revision An entry in RevHistory, describing some revision made to the text. It contains, in order, a required RevNumber and required Date, then, optionally, one or more sets of AuthorInitials, and a RevRemark. It has common attributes.

RevNumber The number of a Revision. It contains plain text and has common attributes.

RevRemark An element of Revision, describing the Revision. It contains plain text and has common attributes.

Row A row in a TBody, THead, or TFoot. It contains one or more Entries or EntryTbls, in any order. It has common, Rowsep, and VAlign attributes.

Rowsep determines Row separators. If its content is 1 (yes), display the internal vertical row ruling below each Row; if 0 (no), do not display it.

It is ignored for the last Row of the TGroup, THead, or TFoot, where the frame value applies. There is no default. The value is inherited from TGroupStyle, if used.

Valign governs the vertical positioning of text within a Row. Allowed values are Top, Middle, and Bottom (no default).

Screen Intended to represent what the user sees or might see on a computer screen. It consists of in-line elements, in which line breaks and leading white space are considered significant. It has common attributes and a Width attribute for specifying a number representing the maximum width of the contents.

ScreenInfo Part of ScreenShot (see *ScreenShot*). A ScreenInfo indicates how the Graphic with which it is paired was created, as a guide for future revisions. It may contain only plain text, and has common attributes.

ScreenShot Like Screen, intended to represent what the user sees or might see on a computer screen. It consists of an optional ScreenInfo and a required Graphic. It has common attributes.

Secondary A word or phrase in the text that is to appear in the index beneath a Primary entry. It must be nested within IndexTerm tags and must follow a Primary element. It may contain in-line elements, and has SortAs and common attributes. See *Primary.*

SecondaryIE Part of IndexEntry, like PrimaryIE (see *PrimaryIE*).

Sect1 A top-level section of a book component, including the Title of that section. Sect2–5 nest in order within Sect1. Anything may occur within a Sect1 except a DocInfo, Preface, Chapter, Appendix, or another Sect1, including a Glossary, Bibliography, RefEntry, ToC, Index, or LoT. A Sect must have a Title, which is the text of the heading itself, and may have a TitleAbbrev; it must include some content, whether paragraphs or other block-oriented elements, including a Sect2. Sect1–5 have common, Label, and Renderas attributes. The Renderas attribute may have the value Sect1 through Sect5, and may be used to specify that the Sect (particularly its heading) is to be presented in the format defined for a Sect of some other level. The default is IMPLIED.

Sect2 A section beginning with a second-level heading; must be nested within a Sect1. Allowable and required contents for Sect2, and its attributes, are like those for Sect1.

Sect3 See *Sect1, Sect2.*

Sect4 See *Sect1, Sect2.*

Sect5 See *Sect1, Sect2.* Sect5 may not contain Sects of any level, and no further subdivisions are supplied in the DocBook DTD.

See Part of IndexTerm, indicating, for a word or phrase in the text, the index entry to which the reader is to be directed when he consults the stub index entry for another element within the IndexTerm. See must be nested within IndexTerm tags and must follow a Primary or Secondary element. It may contain in-line elements, and has common attributes.

SeeAlso Like See, but indicates the index entries to which the reader is *also* to be directed when he consults a full index entry. SeeAlso must be nested within IndexTerm tags and must follow a Primary or Secondary element. It may contain in-line elements, and has common attributes.

SeeAlsoIE A "see also" entry in an Index, not in the text, occurring unnested within IndexEntry at the PrimaryIE or SecondaryIE level. It may contain plain text and in-line elements. It has common attributes and an optional Linkends attribute, which may have the value of some list of IndexEntry IDs.

SeeIE A "see" entry in an Index, not in the text, occurring unnested within IndexEntry at the PrimaryIE or SecondaryIE level. It may contain plain text and in-line elements. It has common attributes and an optional Linkend attribute, which may have the value of some IndexEntry ID.

Seg A component of a SegmentedList. Segs are the only content of a SegmentedList's SegListItems. Seg may contain in-line elements. It has common attributes.

SegListItem A list item in a SegmentedList. It consists of two or more Segs, and has common attributes.

SegmentedList A list of sets of units. It may be used to represent sets of information often presented as simple tables. SegmentedList may have a Title and TitleAbbrev, followed by any number of SegTitles, and one or more SegListItems. It has common attributes.

SegTitle A title that pertains to one Seg in each SegListItem: the first SegTitle to the first Seg, the second SegTitle to the second Seg, and so on. It may contain in-line elements. SegTitles are grouped at the beginning of a SegmentedList, before the SegListItems. It has common attributes.

SeriesInfo Part of BookInfo or BiblioEntry, containing information about the publication series of which the book is a part. SeriesInfo contains, in order, a required Title, optional TitleAbbrev and Subtitle; any number of AuthorGroups, optional ISBN, VolumeNum, and IssueNum; a required SeriesVolNums, any number of PubDates and Publishers, and finally an optional Copyright. It has common attributes.

SeriesVolNums The numbers of all the volumes in a Series, for use in SeriesInfo. It contains plain text (e.g., 1–5) and has common attributes.

Set Two or more Books. Set may have, in order, a Title, TitleAbbrev, SetInfo, and ToC, followed by the Books, followed by an optional SetIndex. Note that a SetIndex may appear in a Book, too. Set has common attributes.

SetIndex Index to a Set. It may occur within a Set or one of the Books in a Set. It contains an optional DocInfo, Title, and TitleAbbrev, followed by any number of block-oriented elements and then one or more IndexEntries or one or more IndexDivs. It has common attributes.

SetInfo Metainformation for a Set, in which it may appear. It may contain, in any order, any number of: Author, AuthorInitials, Copyright, CorpAuthor, CorpName, Date, Editor, Edition, InvPartNumber, ISBN, LegalNotice, OrgName, OtherCredit, PrintHistory, ProductName, ProductNumber, Publisher, PubsNumber, ReleaseInfo, RevHistory, Title, Subtitle, and VolumeNum. SetInfo has common attributes and a Contents attribute. Contents is partly a stub for future development; at the moment its values should be the IDs of the ToC, Books, and SetIndex that comprise the Set, in the order of their appearance.

SGMLTag An element tag in SGML. This element contains plain text, and should not include delimiting angle brackets, ampersands, percent signs, or semicolons: those should be supplied by the renderer if appropriate. It has common and Class attributes. Class may be Attribute, Element, GenEntity, or ParamEntity; there is no default.

ShortAffil Brief version of of Affiliation, in which it may appear. It contains plain text and has common attributes.

Sidebar Segment of a book component that is isolated from the narrative flow of the main text, typically boxed and floating. Sidebar may have a Title and a TitleAbbrev, followed by paragraphs, lists, and other block-oriented elements. No Sects allowed. Sidebar has common attributes.

SimPara A paragraph that is only a text block, without included block-oriented elements. It may contain InlineGraphic, InlineEquation, and synopses. SimPara has common attributes.

SimpleList Intended for long lists of single words or short phrases. It consists of one or more Members, and has common, Columns, and Type attributes. The value of the Type attribute may be Inline, Horiz, or Vert (the default), indicating that the list should be formatted as part of a regular paragraph, as an array reading L to R then top to bottom, or as an array reading top to bottom then L to R. The Columns attribute value must be a number, the number of columns the array should contain. Further details on rendering may be found in the comment in the DTD, but these details may required some revision in the future.

SpanSpec Formatting information for a spanned column in a TGroup (part of Table). SpanSpec is an empty element, bearing common, Align, Char, Charoff, Colsep, Nameend (required), Namest (required), Rowsep, and Spanname, (required) attributes. SpanSpec identifies a horizontal span of columns and associated attributes that can subsequently be referenced by its SpanName to provide attributes repeatedly used in the Entries or EntryTbls in the Rows of the TGroup that is nested within the THead, TFoot, or TBody in which the SpanSpec occurs.

The reason Colname is used rather than Colnum in identifying SpanSpec is that the names are independent of revisions that may change the number of inserted/deleted columns, as long as Namest remains to the left of (has a smaller colnum than) Nameend. SpanSpecs set on THead or TFoot override those on the containing TGroup and apply to just the THead or TFoot. SpanSpecs from the containing TGroup apply to TBody.

Align controls the horizontal position of text within the column. The value of Align may be Left (quad flush left), Center (centered), Right (quad flush right), Justify (both quad left and right), or Char (align to the left of Char, positioned by Charoff). The default is Center.

Char contributes to Align. If the value of Align is "char", the value of Char should be a character on the first occurrance of which the entry is to be aligned. If that character does not occur in the entry, the entry is aligned to the left of the position determined by Charoff. The default is inherited from the ColSpec of the column named by Namest.

Charoff contributes with Char to Align. Charoff is the proportion of the current column width, expressed in percentage, to be allowed before the left edge of the first occurrence of the character given as the value of Char, if any. The default is inherited from the ColSpec of the column named by Namest.

Colsep determines column separators. If its value is 1 (yes), display the internal column rulings to the right of each item; if 0 (no), do not display it. It is ignored for the last column, where the Frame setting applies. (In CALS, 1 (yes) is expressed as 1 and No as 0.) The default is inherited from the ColSpec of the column named by Namest.

Nameend is the name of the rightmost column of the span. Names are identified in the Colspec of the current TGroup.

Namest or Name Start, is the name of the leftmost column of the span. Names are identified in the Colspec of the current TGroup.

Rowsep determines Row separators. If its content is 1 (yes), display the internal vertical Row ruling below each item; if 0 (no), do not display it. It is ignored for the last Row of the table, where the frame value applies. There is no default. The value is inherited from the ColSpec of the column named by Namest.

Spanname is the name of the horizontal span.

State Part of Address. It contains plain text and has common attributes.

Step Part of a Procedure. After its optional Title, Step may contain block-oriented elements mixed with occurrences of the SubSteps element. SubSteps then contains one or more Steps—it is a wrapper. Step has common attributes and a Performance attribute, which indicates whether the step must be performed: the values are Optional and Required (the default).

Street Part of Address. It contains plain text and has common attributes.

StructField A field in a Structure. It contains plain text and has common attributes.

StructName The name of a Structure. It contains plain text and has common attributes.

SubSteps A wrapper for Steps within Steps. See *Procedure, Step.* Note that SubSteps, like Step, has a Performance attribute, the values of which may be Optional or Required (the default).

Subscript A subscript. It may contain basic in-line elements and Replaceable, and has common attributes.

Subtitle Subtitle of a document. It may contain in-line elements.

Superscript A superscript. It may contain basic in-line elements and Replaceable, and has common attributes.

Surname (Western-style) family name of an author. It contains plain text and has common attributes.

Symbol A name that is replaced by a value before processing. It contains plain text and has common attributes.

Synopsis The syntax of a command or function. It appears within RefSynopsisDiv, and may occur elsewhere in a document, too. Synopsis has an optional title, followed by one or more in-line elements and Graphics; within it, line breaks and white space are significant. It has common attributes. CmdSynopsis and FuncSynopsis are alternates to Synopsis, but they may not contain Graphics and within them line breaks and white space are not significant. Synopsis should be displayed as a block-oriented element, not in-line.

SynopFragment Part of CmdSynopsis. It contains one or more Args or Groups, in any order, and has the attributes ID (the only required attribute), Lang, Remap, Role, and XRefLabel.

SynopFragmentRef Part of a CmdSynopsis. It contains RCDATA (characters, in which entity references and character references are recognized in parsing) rather than elements. It has common and Linkend (required) attributes. Linkend should point to SynopFragment.

SystemItem Any system-related item. It may contain members of cptrphrase.gp, and has common, Class, and MoreInfo attributes. Class may have the value Constant, EnvironVar, Macro, OSname, Prompt, Resource, or SystemName (no default). For the MoreInfo attribute see *Application.*

Table An array of text. Table is modelled on CALS tables. The present Table element has a required Title, an optional TitleAbbrev, and either one or more Graphics or one or more TGroups. Tables may not contain either Tables or InformalTables. Table has common, Colsep, Frame, Label, Orient, Pgwide, Rowsep, Shortentry, Tabstyle, and Tocentry attributes. elements that may be contained within TGroup are specified as having omissible end tags, except for EntryTbl. Who said the military isn't thoughtful?

Colsep determines column separators. If its value is 1 (yes), display the internal column rulings to the right of each item; if 0 (no), do not display it. It is ignored for the last column, where the value of Frame applies. There is no default; the value may be inherited from Tabstyle in the FOSI.

Frame describes the positioning of framing rules around the Table. The allowed values are Sides (left and right), Top (below title), Bottom (after last Row, possibly the last Row of TFoot), Topbot (both top and bottom), All (all of the aforementioned), and None. There is no default. The value may be inherited from Tabstyle in the FOSI.

Label is just the same as DocBook's Label attribute: a reference string, such as a number, to be prefixed to the Title.

Orient determines the orientation of the Table. The allowed values are Port (the orientation of the rest of the text of the Book, that is, upright) and Land (90 degrees counterclockwise from the orientation of the rest of the text of the Book). If you assume that the page is taller than it is wide, tables that take up a whole page and are wider than they are tall should be given the Land value. There is no default, the value may be inherited from the FOSI.

Pgwide constrains the table to or liberates it from the boundaries of its column. If the value is 1 (yes), the table runs across the entire page; if 0 (no), the table "runs across just the (galley) width of the current column of the page, regardless of the value of Orient. If the value is 0 (no), it has no meaning for the case when Orient is set to Land. The value may be inherited from Tabstyle in the FOSI, if available.

Rowsep determines row separators. If its content is 1 (yes), display the internal vertical row ruling below each item; if 0 (no), do not display it. It is ignored for the last row of the table, where the value of Frame applies. There is no default. The value may be inherited from TabStyle.

Shortentry determines whether value of ShortEntry is be used in the an Index or ToC or LoT. If the value is 0 (no), the TitleAbbrev is not used; if 1 (yes), the ShortEntry is used. *Usage Note:* Be aware that this mechanism duplicates the DocBook element TitleAbbrev. It would be better, for any given work, to choose one mechanism or the other, rather than mix them.

Tabstyle is the name of a table style defined in the FOSI. There is no default.

Tocentry determines whether the Table's Title should be included in an LoT. The value may be 1 (yes) (include) or No (exclude). The default is 1 (yes) (include). should be ignored if the optional Title is omitted; that would be the case if you were using Table, instead of InformalTable, for a Table that has no Title; presumably only a reference string could then appear in an LoT.

Term The hanging term attached to a ListItem within a VarListEntry in a VariableList; visually, a VariableList is a set of Terms with attached items such as paragraphs. Each ListItem may be associated with a set of Terms. Term may contain in-line elements or synopses. It has common attributes.

Tertiary A word or phrase that is to appear in the index under a Secondary entry. It must be nested within IndexTerm tags and must follow a Secondary element. It may contain in-line elements, and has SortAs and common attributes. See *Primary.*

TertiaryIE Part of IndexEntry, like PrimaryIE see *PrimaryIE*).

TBody Wrapper for the Rows of a Table or InformalTable. It contains simply one or more Rows, and has common and VAlign attributes. VAlign governs the vertical positioning of text within a Row. Allowed values are Top (the default), Middle, and Bottom.

TFoot TFoot is a optional part of TGroup (part of Table), identifying the footer information in a Table, which is displayed after the TBody and also at the bottom of any TBody Rows before a page break. TFoot must have one or more Rows. It has common and VAlign attributes, the latter with the allowed values of Top (the default), Middle (approximately centered vertically), and Bottom.

TGroup The part of a Table that is contains an array along with its formatting information. In order, a TGroup has any number of ColSpecs, any number of SpanSpecs, an optional THead, a required TBody, and an optional TFoot. Each TGroup effectively identifies a new portion of a Table. If a new ColSpec is provided, it replaces a previous one. If both ColSpec and SpanSpec are new, that SpanSpec should refer to columns in the most recent ColSpec. If only a new SpanSpec is provided, it should refer to columns defined by the most immediately prior ColSpecs in a TGroup of the Table. On the other hand, a new ColSpec to either a THead or TFoot replaces all prior column definitions. TGroup has common, Align, Char, Charoff, Cols, Colsep, Rowsep, and TGroupStyle attributes.

Align controls the horizontal position of text within the column. The value of Align may be Left (quad flush left), Center (centered), Right (quad flush right), Justify (both quad left and right), or Char (align to the left of Char, positioned by Charoff). The default is Left, unless overridden by TGroupStyle.

Char contributes to Align. If the value of Align is "char", the value of Char should be a character on the first occurrance of which the entry is to be aligned. If that character does not occur in the entry, the entry is is aligned to the left (the original doc incorrectly specifies "right") of that character. The default is "", unless inherited from TGroupStyle.

Charoff contributes with Char to Align. Charoff is the proportion of the current column width, expressed in percentage, to be allowed before the left edge of the first occurrence of the character given as the value of Char, if any. The default is 50 unless overridden by TGroupStyle.

Cols is the number of columns in the table (required).

Colsep determines column separators. If its value is 1 (yes), display the internal column rulings to the right of each item; if 0 (no), do not display it. It is ignored for the last column, where the value of Frame applies. There is no default. The value is inherited from TGroupStyle, if used.

Rowsep determines row separators. If its content is 1 (yes), display the internal vertical row ruling below each item; if 0 (no), do not display it. It is ignored for the last row of the table, where the value of Frame applies. There is no default. The value is inherited from TGroupStyle, if used.

TGroupStyle is a unique table group style defined in a FOSI (no default).

THead THead is a optional part of TGroup (part of Table). THead may have any number of ColSpecs, followed by one or more required Rows. It has common and VAlign attributes, the latter with the allowed values of Top, Middle, and Bottom (the default). THead identifies the heading information in a Table, which is displayed at the top of the Table and again at the top of any continuation after a page break between Rows in TBody.

Tip A suggestion to the user, set off from the text. See *Caution.*

Title The text of a heading or the title of a block-oriented element. Title may contain in-line elements, and has common and PageNum attributes.

TitleAbbrev An optional, abbreviated version of any Title. You may want to use this element when a title is so long that it might be truncated in some part of an online display, such as a title bar. TitleAbbrev may contain in-line elements and has common attributes.

ToC A Table of Contents, which may be a book component on its own or may occur within other book components. It may have a DocInfo, Title, and

TitleAbbrev. ToC is subdivided to follow the divisions of a book: following the optional Title and TitleAbbrev, a ToC may have any number of ToCFronts, which are the entries for the front matter. Following the ToCFronts, if any, ToC must have either one or more ToCParts (entries for Parts) or ToCChaps (entries for Chapters and Appendices), and may have any number of ToCBacks (entries for back matter). A ToCPart must begin with one or more ToCEntries, then contains any number of ToCChaps. A ToCChap must begin with one or more ToCEntries, then may have any number of ToCLevel1s, which are entries for Sect1s. ToCLevel1s must begin with one or more ToCEntries, then may have any number of ToCLevel2s, and so on down to ToCLevel5, which may have only ToCEntries. Thus if you have a Table of Contents that shows section headings, the second-level entries are nested within the first-level entries, and so on. ToC has common attributes.

ToCBack Entry for back matter in a ToC. See *ToC.* It has common, Pagenum, and Label attributes. The Label attribute might take the value of Bib, Gloss, or Index, depending on the identity of the piece of front matter concerned, but these are not defined.

ToCChap See *ToC.*

ToCEntry Component of other ToC elements. See *ToC.* It has common, Linkend, and Pagenum attributes. PageNum is for indicating the page numbers on which the Table of Contents entries appear in a printed book. Linkend, should you choose to use it, should have the value of the ID of the relevant book part.

ToCFront Entry for introductory matter in a ToC. See *ToC.* It has common, Pagenum, and Label attributes. The Label attribute might take the value of Equations, Examples, Figures, Preface, or Tables, depending on the identity of the piece of front matter concerned, but these are not defined.

ToCLevel1 The top-level tag for entries within a ToCChap. See *ToCChap.* ToCLevel2–5 may be nested within it, in order. All have common attributes. Aside from nested ToCLevels, ToCLevel1–5 must contain one or more ToCEntries.

ToCPart See *ToC.*

Token A unit of information in the context of lexical analysis. It contains plain text and has common attributes.

Trademark A trademark. It may contain members of cptrphrase.gp, and has common and Class attributes. Class may have the values Service, Trade, Registered, or Copyright; the default is Trade.

Type Indicates the classification of a value. It contains plain text and has common attributes.

ULink A link employing a URL. ULink may contain in-line elements. A URL is a Uniform Resource Locator, as used with the World Wide Web. Very generally, a URL is a string specifying, according to a defined protocol, a method, a path to a target, and possibly some additional information. The URL should be given as the value of the URL attribute; there is also a Type attribute, to which you may assign any value.

UserInput Data entered by the user. It may contain elements from cptr-phrase.gp, and has common and MoreInfo attributes. For the MoreInfo attribute see *Application.*

VarArgs An empty element, part of FuncSynopsis, indicating that the Function in question has a variable number of arguments. The string "(...)" should be output.

VariableList An optionally titled list of VarListEntries, which are composed of sets of one or more Terms with associated ListItems; ListItems contain paragraphs and other block-oriented elements in any order. Inclusions are as for OrderedList (see *OrderedList*). VariableList has common attributes.

VarListEntry A component of VariableList (see *VariableList*). It has common attributes.

Void An empty element, part of FuncSynopsis, that indicates that the Function in question takes no arguments. The string "(void)" should be output. See *VarArgs.* It has common attributes.

VolumeNum The number of a Book in relation to Set, or of a journal, when Book is used to represent a journal by containing Articles. It contains plain text and has common attributes.

Warning An admonition set off from the text. See *Caution.*

WordAsWord A word (or letter or number) used not to represent the thing or idea it usually represents, but merely as the word itself. For example, "The term ¡WORDASWORD¿Gothic¡/WORDASWORD¿ means different things to art historians and typographers," or for a single character, "the letter ¡WORDASWORD¿X¡/WORDASWORD¿". It contains plain text and has common attributes.

XRef Cross reference link to another part of the document. It has Linkend and Endterm attributes, just like Link, but like Anchor, it may have no content. XRef must have a Linkend, but the Endterm is optional. If it is used, the content of the element it points to is displayed as the text of the cross reference; if it is absent, the XRefLabel of the cross-referenced object is displayed. XRef also has common attributes. See *Link.*

Year Year of publication, for use in Copyright. It contains plain text and has common attributes.