Internal Document

Open Group Technical Publications Writing Style

The Open Group

Internal Document

Open Group Technical Publications Writing Style

ISBN: N/A
Document Number: I801A (Writing Style)

Published in the U.K. by The Open Group, December 1998.

Any comments relating to the material contained in this document may be submitted to:

The Open Group
Apex Plaza
Forbury Road
Reading
Berkshire RG1 1AX

Or by email to:

OGEdit@opengroup.org

Contents

Preface

The Open Group

The Open Group is the leading vendor-neutral, international consortium for buyers and suppliers of technology. Its mission is to cause the development of a viable global information infrastructure that is ubiquitous, trusted, reliable, and as easy-to-use as the telephone. The essential functionality embedded in this infrastructure is what we term the *IT DialTone*. The Open Group creates an environment where all elements involved in technology development can cooperate to deliver less costly and more flexible IT solutions.

Formed in 1996 by the merger of the X/Open Company Ltd. (founded in 1984) and the Open Software Foundation (founded in 1988), The Open Group is supported by most of the world's largest user organizations, information systems vendors, and software suppliers. By combining the strengths of open systems specifications and a proven branding scheme with collaborative technology development and advanced research, The Open Group is well positioned to meet its new mission, as well as to assist user organizations, vendors, and suppliers in the development and implementation of products supporting the adoption and proliferation of systems which conform to standard specifications.

With more than 200 member companies, The Open Group helps the IT industry to advance technologically while managing the change caused by innovation. It does this by:

- Consolidating, prioritizing, and communicating customer requirements to vendors
- Conducting research and development with industry, academia, and government agencies to deliver innovation and economy through projects associated with its Research Institute
- Managing cost-effective development efforts that accelerate consistent multi-vendor deployment of technology in response to customer requirements
- Adopting, integrating, and publishing industry standard specifications that provide an essential set of blueprints for building open information systems and integrating new technology as it becomes available
- Licensing and promoting the Open Brand, represented by the "X" Device, that designates vendor products which conform to Open Group Product Standards
- Promoting the benefits of the IT DialTone to customers, vendors, and the public

The Open Group operates in all phases of the open systems technology lifecycle including innovation, market adoption, product development, and proliferation. Presently, it focuses on seven strategic areas: open systems application platform development, architecture, distributed systems management, interoperability, distributed computing environment, security, and the information superhighway. The Open Group is also responsible for the management of the UNIX trademark on behalf of the industry.

Development of Product Standards

This process includes the identification of requirements for open systems and, now, the IT DialTone, development of Technical Standards (formerly CAE and Preliminary Specifications) through an industry consensus review and adoption procedure (in parallel with formal standards work), and the development of tests and conformance criteria.

This leads to the preparation of a Product Standard which is the name used for the documentation that records the conformance requirements (and other information) to which a vendor may register a product.

The ‘‘X’’ Device is used by vendors to demonstrate that their products conform to the relevant Product Standard. By use of the Open Brand they guarantee, through the Open Brand Trade Mark License Agreement (TMLA), to maintain their products in conformance with the Product Standard so that the product works, will continue to work, and that any problems will be fixed by the vendor.

Open Group Publications

The Open Group publishes a wide range of technical documentation, the main part of which is focused on development of Technical Standards and product documentation, but which also includes Guides, Snapshots, Technical Studies, Branding and Testing documentation, industry surveys, and business titles.

There are several types of specification:

- *Technical Standards* (formerly *CAE Specifications*)

  The Open Group Technical Standards form the basis for our Product Standards. These Standards are intended to be used widely within the industry for product development and procurement purposes.

  Anyone developing products that implement a Technical Standard can enjoy the benefits of a single, widely supported industry standard. Where appropriate, they can demonstrate product compliance through the Open Brand. Technical Standards are published as soon as they are developed, so enabling vendors to proceed with development of conformant products without delay.

- *CAE Specifications*

  CAE Specifications and Developers’ Specifications published prior to January 1998 have the same status as Technical Standards (see above).

- *Preliminary Specifications*

  Preliminary Specifications have usually addressed an emerging area of technology and consequently are not yet supported by multiple sources of stable conformant implementations. They are published for the purpose of validation through implementation of products. A Preliminary Specification is as stable as can be achieved, through applying The Open Group’s rigorous development and review procedures.

  Preliminary Specifications are analogous to the *trial-use* standards issued by formal standards organizations, and developers are encouraged to develop products on the basis of them. However, experience through implementation work may result in significant (possibly upwardly incompatible) changes before its progression to becoming a Technical Standard. While the intent is to progress Preliminary Specifications to corresponding Technical Standards, the ability to do so depends on consensus among Open Group members.

- *Consortium and Technology Specifications*

  The Open Group publishes specifications on behalf of industry consortia. For example, it publishes the NMF SPIRIT procurement specifications on behalf of the Network Management Forum. It also publishes Technology Specifications relating to OSF/1, DCE, OSF/Motif, and CDE.

  Technology Specifications (formerly AES Specifications) are often candidates for consensus review, and may be adopted as Technical Standards, in which case the relevant Technology Specification is superseded by a Technical Standard.

In addition, The Open Group publishes:

- *Product Documentation*

  This includes product documentation—programmer's guides, user manuals, and so on—relating to the Pre-structured Technology Projects (PSTs), such as DCE and CDE. It also includes the Single UNIX Documentation, designed for use as common product documentation for the whole industry.

- *Guides*

  These provide information that is useful in the evaluation, procurement, development, or management of open systems, particularly those that relate to the Technical Standards or Preliminary Specifications. The Open Group Guides are advisory, not normative, and should not be referenced for purposes of specifying or claiming conformance to a Product Standard.

- *Technical Studies*

  Technical Studies present results of analyses performed on subjects of interest in areas relevant to The Open Group's Technical Program. They are intended to communicate the findings to the outside world so as to stimulate discussion and activity in other bodies and the industry in general.

Versions and Issues of Specifications

As with all *live* documents, Technical Standards and Specifications require revision to align with new developments and associated international standards. To distinguish between revised specifications which are fully backwards compatible and those which are not:

- A new *Version* indicates there is no change to the definitive information contained in the previous publication of that title, but additions/extensions are included. As such, it *replaces* the previous publication.

- A new *Issue* indicates there is substantive change to the definitive information contained in the previous publication of that title, and there may also be additions/extensions. As such, both previous and new documents are maintained as current publications.

Corrigenda

Readers should note that Corrigenda may apply to any publication. Corrigenda information is published on the World-Wide Web at *http://www.opengroup.org/corrigenda*.

Ordering Information

Full catalogue and ordering information on all Open Group publications is available on the World-Wide Web at *http://www.opengroup.org/pubs*.

Preface

This Document

This document defines The Open Group writing style for technical publications.

It should be followed to ensure a unified approach to documentation style, organization, terminology, and appearance.

This document is intended for anyone who drafts or writes technical documents for publication by The Open Group.

Submissions to The Open Group that conform to this house style can be processed more quickly than those that do not.

Readers are expected to be familiar with text editors, word processors, and the American-English language. Readers should also understand the principles of text processing using formatting commands. Detailed knowledge of text processors used by The Open Group is not required.

Trademarks

Motif®, OSF/1®, UNIX®, and the "X Device"® are registered trademarks and IT DialTone™ and The Open Group™ are trademarks of The Open Group in the U.S. and other countries.

POSIX® is a registered trademark of the Institute of Electrical and Electronic Engineers, Inc.

PostScript® is a registered trademark of Adobe Systems Incorporated.

Referenced Documents

The following documents are referenced in this guide:

Read Me First! A Style Guide for the Computer Industry, SunSoft Press, A Prentice Hall Title, 1996, ISBN 0-13-455347-0.

The Chicago Manual of Style, University of Chicago Press.

ISO 8859-1:1987, Information Processing — 8-bit Single-byte Coded Graphic Character Sets — Part 1: Latin Alphabet No. 1.

Webster's Collegiate Dictionary, Merriam-Webster.

Chapter 1

Document Structure

This chapter describes the overall structure of technical documents for publication by the Open Group.

The structure of technical documents published by The Open Group is based upon the guidelines contained in *Read Me First! A Style Guide for the Computer Industry*.

The presentation of information about system items is based on the layout of UNIX reference pages.

1.1 Front Matter

1.1.1 Title Page

The type of document and its title, and the text "The Open Group."

1.1.2 Copyright Page

Copyright information (see Section 2.8 on page 8), together with the type of document, its title, ISBN and document numbers, and contact details for problem reporting.

1.1.3 Contents

Lists parts, chapters, sections (second-level headings), subsections (third-level headings), reference pages, appendixes, glossary, and index.

1.1.4 Preface

1. Introduction

   About The Open Group and its document types.

   The same text is used in all Open Group technical documentation and is supplied by The Open Group.

2. This Document

   A brief introduction to the document and its purpose.

3. Intended Audience

   Readership and prerequisite knowledge.

4. Structure

   Brief one-line descriptions of the content of each element.

5. Applicability (Optional)

Version of software documented.

6. Typographical Conventions

   Use of fonts and special symbols.

7. History (Optional)

   How this document relates to others.

8. Problem Reporting (Optional)

   Describes how to report problems with the software documented. Provides a contact and method for problem reports or suggestions for improvement.

1.1.5 Trademarks

A full list of trademark acknowledgements for all trademarks used in the document.

1.1.6 Acknowledgements

A list of contributors.

1.1.7 Referenced Documents

A list of all documents referenced with full bibliographic details. The details should include the title, version number (if applicable), author, publisher, ISBN number, document number (if applicable), and date of publication.

A list of related documents can also be included, introduced by text such as: ‘‘The following documents are not necessary for implementation of the specification, but provide additional information likely to be of value to implementors or application writers.’’

1.2 Body Text

Body text is made up of the following elements: chapters, reference pages, appendixes, a glossary, and an index. Each element is a separate file, although these may in turn be made up from other files.

Each element can include paragraphs, tables, figures, lists, notes, footnotes, and cross-references.

1.2.1 Chapters

Each chapter is an element of a document.

Each chapter covers one main topic. The definition of a topic varies according to the subject matter of the document. Each chapter may consist of numbered sections (second-level headings) and subsections (third-level headings). Fourth-level headings are discouraged. Fifth-level and lower headings must not be used.

The first chapter of a document should include the following:

- Objective or Purpose

  This is a brief statement.

- Overview

This is an overview of the subject.

- Conformance

  This identifies expected criteria for conformance to existing standards or the Open Brand. It is required in Technical Standards. In all cases, it is for guidance only; definitive conformance requirements are given in branding documentation.

- Future Directions

  This describes how the publication and related publications are expected to develop.

If there are sections or subsections, there must be more than one. A section should have subsections if the subject matter is subordinate to the main theme of the section.

If the restrictions on sections and subsections cannot easily be implemented, consider modifying the structure of the document to obtain a suitable structure within each chapter.

Unnumbered headings can be used to aid readability; they do not appear in the table of contents.

1.2.2 Reference Pages

Reference pages have a special layout (see Chapter 3 on page 23). Each page may be contained in its own file, in a chapter, or in an appendix.

Any reference page section must be preceded by a section heading and introductory text. The pages should be sorted into alphabetical order.

1.2.3 Appendixes

Each appendix is an element of a document.

An appendix contains material that is required for reference, but that would interrupt the flow of information in a chapter; for example, long tables of data. Appendixes may contain material that is normative or non-normative; in cases where it is normative, this will be clearly stated.

An appendix may contain sections and subsections, like a chapter. The guidelines about the numbers of sections and subsections are the same as for chapters.

An appendix can also contain reference pages.

Information in appendixes can include, but is not limited to, the following:

- Descriptions of data formats and file structures
- Input and output codes; for example, character conversion codes
- Global processing limitations
- Sample files, reports, or programs

1.2.4 Glossary

Brief definitions of terms used in the document. This is optional (though preferred), but is mandatory in Technical Standards.

1.2.5 Index

This is mandatory in all documents.

Chapter 2

Writing Style

The Open Group preferred writing style is based upon the guidelines contained in *Read Me First! A Style Guide for the Computer Industry*.

The information presented here summarizes the key features of this writing style, and documents decisions made for areas where the above guide does not recommend specific editorial policy.

This chapter starts with a list of general writing style guidelines, and thereafter consists of an alphabetically ordered list of style components.

2.1 General Guidelines

These general guidelines are designed to take account of internationalization and translation considerations:

- Sensitivity to your readers' needs is important. It is necessary to start with a good understanding of who your readers are likely to be, and their technical expertise. This should be clearly identified in the Intended Audience section of the Preface.
- Anticipate the readers' questions.
- Provide information clearly and concisely, so that readers can find information quickly.
- Keep sentences short—preferably less than twenty five words (approximately one and one half lines)— and clear (but complete).
- Use consistent language, terminology, and typographical conventions. Try to avoid the use of synonyms.
- Make sure statements are unambiguous—do not use contractions.
- When choices exist, explain the advantages and disadvantages of the alternatives.
- Avoid humor, jargon, irony, idioms, adages, slang, sexist language, and political and religious references.

  If necessary, when using computer terms that can be interpreted as jargon, make sure that you and the reader understand the meaning of the word as used in your document. Ideally such terms should be included in the Glossary.
- Define key terms at the first mention, if the terms may be new to the reader. (And add to the glossary.)
- Avoid general modifiers, such as ‘‘nice.’’
- Avoid long strings of modifiers. For example, ‘‘The previously sent destination protocol address ...’’ would be better worded as ‘‘The destination protocol address that was previously sent ...’’

- Do not use the same word in different grammatical categories.
- Avoid using symbols to represent words, such as an ampersand (&) to represent the word "and."
- Avoid referring to the authors. If such a reference is essential, use "The Open Group ..."
- Use *x* to refer to a generic letter, and *n* to refer to a generic number.
- Do not use double spaces between sentences.
- Do not refer to holidays.
- Do not use analogies and terms based on local culture.
- Do not use non-English abbreviations and terms (see Appendix A on page 55).
- Do not use terms that attribute human characteristics to software, including gender.
- Do not use words that do not appear in the dictionary (see Section 2.33 on page 19).

2.2 Abbreviations, Acronyms, and Mnemonics

Where possible, keep to commonly used abbreviations, acronyms, and mnemonics and avoid inventing new ones. Once introduced, use them consistently.

An abbreviation is a shortened form of a term. An acronym is a word formed from the initial letters or parts of compound terms. A mnemonic is a memory aid.

For clarity, define abbreviations, acronyms, and mnemonics the first time you use them and add them to the glossary. In such cases, show the spelled-out version followed by the abbreviation, acronym, or mnemonic in parentheses. For example:

> the X/Open Transport Interface (XTI)

Use uppercase letters for acronyms and mnemonics. Use uppercase initials for spelled-out versions where this is helpful. Acronyms can include lowercase letters if they are trademarks.

The derivation of an abbreviation, acronym, or mnemonic may be of little importance to the reader if it is commonly used; for example, BASIC or FORTRAN. In this case, omit the spelled-out version.

If the pronunciation of an acronym may not be obvious to the user, provide the pronunciation.

The preceding indefinite article (*a* or *an*) depends on the way the abbreviation is commonly pronounced; *an* is used where the abbreviation begins with a vowel sound, otherwise use *a*. For example, "a mOSI" (pronounced "mosee"), "a LAN," "an NDR."

Do not use an apostrophe to form the plural of an abbreviation, acronym, or mnemonic. Add a lowercase "s;" for example, OEMs.

Avoid beginning sentences with abbreviations, acronyms, or mnemonics, unless they have been fully explained in preceding text.

Do not use periods or intervening spaces with abbreviations, acronyms, and mnemonics unless the abbreviation can be confused with an actual word; for example, "no." for number, or "in." for inch.

Abbreviations may be used in tables, examples, figures, and footnotes.

Use periods in the acronyms of country names; for example, U.K. and U.S.A.

2.3 Acknowledgements

Create a list of contributors, and indicate the nature of their contribution, such as author of source material, provider of source material, drafting, reviewing, and so on.

If applicable, you may include a list of the members of the Working Group involved in the development of the document.

2.4 Alphabetical Order

In general, any list of items which does not have a sequence or priority should be arranged in alphabetical order.

Use the order produced by the *sort* command, except that all special characters should be grouped together.

The following list (read from left to right, from the top down) shows the correct alphabetical sequence:

```
!  ``  #  $  %  &  '  (  )  *  +  '  -  .  /  :
;  <  =  >  ?  @  [  \  ]  ^  _  `  {  |  }  ~
0  1  2  3  4  5  6  7  8  9
A  a  B  b  C  c
 .
 .
 .
X  x  Y  y  Z  z
```

Alphabetize acronyms according to their shortened form.

Use locale-specific collating order for non-English languages.

2.5 Capitalization

Use initial capitals for nouns that refer to a specific person, title, or object.

Do not use initial capitals for nouns that refer to generic or non-specific people, titles, or objects; for example, system manager, computer, program.

Use initial capitals for product names.

Use initial capitals for the names of objects capitalized on the screen; for example, "The Clear menu item ...".

Use initial capitals for the following words when followed by a letter or number: Chapter, Appendix, Section, Version, Release, Guideline, Table, and Figure.

Do not use initial capitals for the words step, line, or column when they are followed by a number.

See also Section 2.37 on page 21.

2.6 Cautions

Use a caution to draw the reader's attention to information which can have a significant or serious impact on the subject being described. See also Section 2.28 on page 16 and Section 2.39 on page 22.

2.7 Contents

The table of contents is produced automatically when a document is built.

2.8 Copyright

All Open Group documents are copyrighted, and this information must be included in all documentation.

Copyright notices must include the word *Copyright* and the copyright symbol (©), the copyright date (month, year), and the copyright owner. For example, "Copyright © July 1997, The Open Group."

Follow this with the words "All rights reserved." on a separate line.

Include The Open Group standard statement concerning restricted rights.

A statement of the country of origin should be added, such as "Published in the U.K, by The Open Group, July 1997."

All copyright information will be verified by The Open Group.

2.9 Cross-References

Cross-references can be made to complete elements (such as chapters and appendices), sections, figures, tables, and examples.

Refer to all figures, tables, and examples *before* they appear.

If a table or figure has no identifier, you can refer to it is as "the following ...", provided this text is immediately before the referenced object.

Do not hard-code page numbers.

All internal cross-references should be as precise as possible, to ensure rapid location of the required material. Note that a reference to Section 7.1 is an instruction to the reader to read all of Section 7.1, not merely the introductory paragraph.

Brief cross-references should be placed in parentheses within sentences. Longer cross-references should be placed in a separate sentence.

2.10 Dates

Dates should be written out in full. The month should be spelled out and the year should not be abbreviated, even when used in a range. For example, "January 5, 1998."

If dates are used in examples, include a comment that explains the purpose, so that it can be translated correctly.

2.11 Equations

Equations should be used with care—they may not display adequately in all output formats.

2.12 Examples

Consider adding examples to illustrate the following:

- Clarification of a point in the text
- Typical usage of a system item (if not obvious)
- Illustration of the differences between related system items
- Complex usage of a system item

Do not use artificial names such as "foo" and "bar" in examples. Try to use meaningful names for file names and other arguments used in examples.

Vary the use of names for people—Anglo-Saxon and non-Anglo-Saxon, male and female. A telephone directory is a useful source of ideas.

In location examples, use cities that are recognizable without the state or country.

Never include names of actual system accounts and passwords. Remember to edit system and user information out of screen-captures.

Program examples should include extensive comments as part of the program text.

Avoid the use of editorial "we", "our", and "let's" in text surrounding examples.

Avoid examples that require an alphabetically ordered list of abbreviations, acronyms, or mnemonics to convey meaning. This can cause translation difficulties because the translated list will probably not be in alphabetical order. If you cannot avoid using an alphabetically ordered list, include a comment in the source file indicating the purpose of the example so that translators can design an example that is appropriate.

Example titles should use the conventions described in Section 2.37 on page 21.

If the example shows an action performed by a system item, use the gerund form of the action in the example title (for example, "Sorting a File"). Do not use gerunds in titles of descriptive examples (such as showing a sample file or presenting a table of useful expressions).

2.13 Extensions

Extensions identify features of particular interest, such as extensions to existing standards or known problems.

The definition of each type of extension is given in Appendix B on page 69.

2.14 External References

The preferred method for referring to any document is to use standard text, such as a string, citation, or text entity.

You can use an abbreviated document title within the text of a document (for example, "see the ANSI COBOL standard"), but you should include the full title and bibliographic details in the Referenced Documents section.

When referring to a book produced by The Open Group, use the book's abbreviated title (for example, "see the XA Specification").

Do not refer to specific elements (such as chapters or sections) of another document.

All bibliographic details should be verified using the following URLs:

Open Group Documents	http://www.opengroup.org/publications/catalog
Standards Documents	http://www.opengroup.org/sib
ISO Catalog	http://www.iso.ch
RFCs	http://info.internet.isi.edu:80/in-notes/rfc/files

The correct information should become part of the document source so that it can be reproduced at any point in the future.

Brief cross-references should be placed in parentheses within sentences. Longer cross-references should be placed in a separate sentence.

2.15 Filenames, Pathnames, and URLs

Use initial periods when referring to a file type. For example, "a .c program," "the .LIS file."

If punctuation characters are part of a filename, pathname, or URL, set it off from the text when its appearance in a sentence might be confusing.

2.16 Font Usage

Refer to Chapter 6 on page 47 for font usage.

2.17 Footnotes

Use footnotes with care to avoid introducing distracting information which can hinder rather than help the reader. If an explanation is required that would interrupt the flow of information in the text, use a footnote.

2.18 Glossary

A glossary should include specialized words, abbreviations, acronyms, and mnemonics used in the body of the document.

Each glossary entry should consist of the term itself, followed by a brief definition. A brief expanded definition of the term may also be added.

Synonyms of words used in the text should be included with a cross-reference to the main glossary entry.

A central Open Group look-up will be made available in due course.

2.19 Grammar

Where no particular rule is specified and if there is a difference between usage in Britain and the U.S., the U.S. version is used.

Use simple syntax, present tense, and active voice whenever possible.

Use imperative verbs to tell the user what to do.

An infinitive (to *verb*) must be treated as one word; do not split the infinitive by inserting another word between the *to* and *verb*.

A sentence must not end with a preposition.

Avoid using irregular perfect participles such as *learnt* and *spelt*, using instead *learned* and *spelled*.

Do not leave out articles, such as ‘‘a’’, ‘‘an,’’ and ‘‘the’’. This style of writing can lead to misinterpretation and incorrect translation. (This rule may be waived where space is limited, such as in a table, figure, or example.)

Make sure the noun to which a pronoun refers is clear. If necessary, repeat the noun.

The following pronouns are always singular: another, either, each, neither, every, one, any, some, anybody, everybody, everything, someone, nobody, nothing, no-one. The following pronouns are always plural: both, others, few, several, many.

Avoid using system items as verbs; for example, use ‘‘use *grep* to find the string’’, rather than ‘‘grep for the string ...’’.

2.20 Graphics

Use an initial capital for the first letter of every word of text used in the graphic, with the exception of flowcharts and data structures (where just the first word is capitalized), and case-sensitive system item names.

Figure titles should use the conventions described in Section 2.37 on page 21.

2.21 Hyphenation

Use the hyphen for the following prefixes:

all- cross- half- multi- non- post- quasi- self-

Do not use the hyphen for the following prefixes:

anti bi co dis extra infra inter intra macro meta micro
mid mis multi non over pre pseudo re sub super ultra un

However, when any prefix is followed by a word beginning with the same letter, add a hyphen.

Prefixes should always include a hyphen if the root word is all uppercase, has an initial capital, a number expressed as a figure, or if the root element is a hyphenated compound.

Use a hyphen if adding a prefix results in unclear meaning. For example, re-collect/recollect, re-cover/recover, re-solve/resolve, and so on.

The words in a noun phrase, when the phrase is functioning as an adjective or modifier, are joined by a hyphen (even though they might not be otherwise). For example, a process initiated by the user is a user-initiated process.

The following suffixes can be added to nouns:

-based -defined -dependent -independent -oriented -specific

The compound adjectives so formed must be used with care. Do not adopt this practice where the sentence is simpler and clearer with a different construction; for example, ‘‘the transaction-manager-dependent issue’’ is clearer as ‘‘the issue dependent on the transaction manager.’’

Adverbial phrases are not hyphenated; for example, ‘‘externally developed program.’’

Hyphenate a modifying phrase when it precedes the noun it modifies. For example, ‘‘state-of-the-art design.’’

2.22 Index

An index is produced automatically when the document is built, provided index terms have been added.

The index should consist of important words and symbols from the text, together with concepts, synonyms, or paraphrases that are related to the main index entries.

Be consistent in spelling index entries and avoid using plurals and initial capitals wherever possible, so that all entries on one topic can be collected together correctly.

Do not use automatic indexing tools. The results are typically too long and do not adequately pinpoint information for the reader.

2.23 Keyboard Keys

When referring to the name of a control key on a keyboard (such as *Control* or *Enter*), use an initial capital letter.

When referring to the name of a key which is not on a keyboard, use a lowercase initial; for example, the space bar, the comma key.

Use *x* to refer to a generic letter key, and *n* to refer to a generic number key.

Refer to a control sequence in text as follows: Ctrl-*x*. That is, use an initial capital for the word "Ctrl", followed by a hyphen, followed by the lowercase letter. Use ˆ*x* to refer to a control sequence when describing system output.

Place angle brackets around key names that are labeled on the keyboard; for example, <Return>.

Use <Return> to refer to the key used to enter commands.

Use the verb *press* when referring to keys.

Refer to *keys* on the keyboard, not *buttons*.

Do not use the name of a key as an adjective. For example, use "Press <Return>" and not "Press the Return key."

Functions may be bound to different keys due to localization. Follow these guidelines when documenting keyboards:

- Select a default keyboard, and develop a method for providing information about alternative keyboards.
- Put as much keyboard information online as possible.
- Use *function names* rather than *key cap names* in documenting software applications.

2.24 Lists

Lists are a good way to break up long sentences or paragraphs and to clarify choices and steps, but it is important that lists are introduced with a clear (but brief) lead-in phrase. Too many lists without clear lead-in phrases interrupt the flow of the text, and can be distracting to the reader.

Use a colon after a lead-in phrase that is a complete statement; for example, “The values of *local* are defined as follows:”.

The type of list you use depends on the type of items contained in the list:

- Use an unnumbered (or unordered) list to show choices or groups of items for which there is no sequence.
- Use a numbered (or ordered) list for items that occur in a specific sequence (such as a procedure or a list of results for some action), for a list that shows the precedence of items, or a hierarchy.
- Use a definition (or variable) list for items that have a term and a corresponding definition or description.

Use the following rules when constructing lists. These rules apply to all types of lists, unless otherwise indicated:

- End each list item with a period if the list item contains a complete sentence. Do not use end punctuation for list items that do not contain a complete sentence.
- Avoid mixing complete sentences and sentence fragments as items in the same list. If you are unable to avoid mixing sentences and sentence fragments, use a period after each list item.
- Start each list item with the same part of speech if possible.
- Capitalize the first word of every list item, whether the item is a complete sentence or not, except for literal names that are lowercase. If possible, reword the list item to avoid this.
- Avoid putting items of obviously dissimilar values in the same list.
- Use parallel construction for all items within a list. For example, lists of options or parameters in reference pages always use the name of each option or parameter as the term of an entry in a definition list. The definition of the term always begins with a verb in the present tense, so that the term and definition, when read together, form a complete sentence. For example:

  –a Lists all information.

  –b Lists only basic information.
- In a procedure, use only complete sentences for each item. Begin each step with a verb in the imperative form, and tell the reader what will happen as a result of each step. Avoid using cross-references in an ordered list. It is better to give all the needed information within a step, rather than referring the reader elsewhere for information needed to complete a task.

Nested lists can be used as follows:

- Unnumbered lists within variable or numbered lists
- Numbered lists within variable lists
- Numbered lists within numbered lists
- Unnumbered lists within unnumbered lists

If a list item runs to several paragraphs, or includes complex material, you should consider using unnumbered headings instead of a list.

For lists in text use commas to separate items in a series of three or more words, phrases, or clauses, including the last item before a conjunction. For example, “a, b, and c.”

2.25 Measurement

Provide measurements as imperial unit symbols, followed by the metric equivalent in parentheses.

Make sure that the precision of a converted measurement reflects the precision of the original measurement.

Place the abbreviation for a unit of measurement at the end of a series of two or more items; for example, “1200, 1400, or 1600MHz.”

Repeat the abbreviation for a unit of measurement in a series of two items; for example, “10 ft to 12 ft.”

Plural and singular abbreviations of units of measurement are the same; for example, “1 lb” and “10 lb.”

Insert a space between a number and the unit of measurement it modifies, except for KB, MB, kHz, MHz, GHz, and K. For example, “6 ft”, and “6MB”.

The context should enable differentiation between the two meanings of *K*: binary thousand and Kelvin.

Avoid using the number sign (#), single quote (’), and double quotes (") symbols to indicate the pound, foot, and inch units of measurement.

Comment source files to indicate which units of measurement are used as an aid to translators.

2.26 Monetary Values

Avoid monetary values since they are country-specific.

Monetary values may be included in examples, but there must be a comment in the source file indicating their purpose as an aid to translators.

2.27 Names

Use initial capitals for names.

The name of an organization is treated as a singular noun. For example, "The Open Group publishes documents."

2.28 Notes

Notes should be used to call attention to important information that the reader should not overlook. They should be used sparingly. See also Section 2.6 on page 8 and Section 2.39 on page 22.

2.29 Numbers

As a general rule, use numerals rather than spelled-out numbers, and be consistent in their use.

Avoid beginning sentences or headings with numerals.

Use *to* (and not *through*) for inclusive ranges. In tables and graphics, use a dash.

Hyphenate numbers or numerals in compound modifiers (500,000-byte file), but not in single modifiers (500,000 bytes).

Use a comma in numerals of more than four digits (for example, 10,000). Do not use commas in binary, octal, or hexadecimal numbers.

Do not use an apostrophe to form the plural of a number. Instead, add a lowercase s; for example, 4s, 1920s.

Insert a space between a number and the unit symbol it modifies, except for degrees, minutes, and seconds of an angle.

2.30 Pagination

Refer to Chapter 6 on page 47 for pagination.

2.31 Punctuation

( ) Parentheses enclose qualifying or explanatory material that is included in a sentence or paragraph.

If the text inside parentheses is a complete sentence, put the terminating period inside the parentheses and terminate the preceding sentence. (This is an example.)

If the text inside parentheses is not a complete sentence, embed it in a sentence, so that the normal period or comma appears on the outside of the parentheses (like this).

If required by the context, place a comma after the closing bracket.

Try to avoid nested parentheses in text; nested tangential thoughts can be difficult to follow. Use several sentences inside the parentheses if necessary.

[ ] Brackets (do not use "square") are not used in text. They are used literally in syntax to indicate bracket characters. Each document should describe how it uses brackets in a

section on ‘‘Typographical Conventions.’’

{ } Braces (do not use ‘‘curly’’) are not used in text. They are used literally in syntax to indicate brace characters. Each document should describe how it uses braces in a section on ‘‘Typographical Conventions.’’

< > Angle brackets are not used in text. They are used literally in some cases where, for example, C programs use them to indicate a filename. In arithmetic, text may refer to these characters as *less than* and *greater than*, respectively.

They should be placed around key names that are labeled on the keyboard; for example, <Return>.

’ An apostrophe indicates possession or a missing character. It is never used to form a plural.

Its is the possessive pronoun of it. *It is* should be used in preference to its contracted form *it’s*.

Use an apostrophe and a lowercase ‘‘s’’ to form the plural of a lowercase letter used as a noun; for example, a’s, y’s.

This character is also a single quote. Use 66-99 quotes in preference to single quotes. However, the single quote may be used literally in syntax.

: A colon directs the reader’s attention to whatever follows it.

Use a colon when you use *for example*, *the following*, *follows*, or *as follows* to lead in to an object beginning on the line below.

Use a colon at the end of lead-in phrases for lists, tables, figures, and so on.

; Use a semicolon to join closely related independent sentences or clauses.

When the elements in a series are long and complex, or involve internal punctuation, they should be separated by semicolons.

(Double quote) Use literally in syntax if required. See 66-99 below.

‘ (Back quote) Use literally in syntax if required.

“ ” (66-99) In text, quote using 66-99 (two back quotes and two single quotes). Only use quotes when the item quoted would look awkward, incomplete, or ambiguous without them, or when the reader would have trouble parsing the sentence without them. Do not use them when defining a new term.

Only use quotes as a way to emphasize a word or phrase when that word or phrase is unique in some context or technical sense.

Use quotes to indicate material quoted from another source, or to enclose single letters.

Do not use quotes where they may be interpreted as part of the syntax. If in doubt, place the text on its own line.

Do not put quotes around items designated as screen objects.

Closing quotation marks (’’) appear after:

- Most adjacent punctuation marks (commas and periods)
- Question marks and exclamation points when the question marks and exclamation points are part of the quoted material

Closing quotation marks appear before:

- Colons
- Semicolons
- Question marks and exclamation points, when the question marks and exclamation points are not part of the quoted material

, (Comma) Place a comma before the conjunction in a compound sentence (consisting of two or more independent clauses), unless the clauses are short and closely related. For example, “The system prints an error message, but you can continue processing the file.”

Use commas to set off a non-restrictive modifier which provides additional information but does not affect the meaning of the words it modifies. For example, “A symbol value may be an absolute constant, expressed as a 32-bit integer, or a relocatable value.” Conversely, do not use commas to set off a restrictive clause which does affect the meaning of the word it modifies. For example, “Table 6-1 describes the hardware that you need to complete your system.”

Use commas to set off contrasting and opposing expressions within sentences. For example, “He changed the software, not the hardware.”

Place a comma after an introductory clause or long introductory phrase. For example, “To specify an output device, enter a name in the command line.”

2.32 Special Characters

The standard character reference used by The Open Group is the ISO 8859-1 Latin-1 character set.

When referring to a printable character for the first time, use the spelled-out name of the character first, followed by the character in parentheses (except in examples, where you should use the character literally). You can add the word “character” if necessary (for example, “the backslash character”). After the first occurrence, you can use the name of the character without showing the character itself.

A special character has no plural form. Spell out the name; for example, “Enter three backslashes (\\\).”

When referring to non-printable characters, use the name of the character alone in text. For example, “You must separate arguments with a space character.” If the character must be shown without spaces between it and an adjacent character, enclose the name of the non-printable character in angle brackets. For example:

```
<Tab><Tab>field1
```

The special characters you are most likely to use are:

#	The international number symbol; referred to as the *hash sign*.
£	The UK pound sign (UKP). It should be referred to as the Sterling sign to avoid ambiguity.
@	The at sign.
&	The ampersand.
*	The asterisk.
\	The backslash.
!	The exclamation mark.
$	The dollar sign.
%	The percent sign.

+ The plus sign.

/ The slash. Use to separate components of a pathname. Do not end pathnames with a slash. Do not use a slash before a single directory or filename unless it refers to the system root. Do not use the standalone slash to refer to the root directory; use *root*.

= The equals sign.

? The question mark.

ˆ The circumflex.

_ The underscore.

` The grave accent.

| The vertical bar or pipe symbol.

˜ The tilde.

— The em-dash. Use to set off an appositive series from the rest of a sentence, to show an abrupt change in thought, and to set off material for emphasis. With the correct use of commas and parentheses, the em-dash is nearly redundant. Do not place spaces around the em-dash. The exception to this rule is in the NAME section of a reference page:

ls — list contents of directory

- The en-dash. Use to indicate a range; for example, 00-61.

– The minus sign. Use to indicate negative numbers. If the minus sign appears in a program display, use the <-> key.

... Ellipses (horizontal or vertical) indicate the omission of information. In syntax, ellipses indicate an item that repeats. If an ellipsis is part of a screen object, include it in the name; for example, ‘‘The Open... menu item.’’ When using an ellipsis in text, put a space before it and use three periods without spaces.

2.33 Spelling

Use American English spelling.

As an authority for spellings not specifically recommended in this guide, see *Webster's New Collegiate Dictionary*. An on-line version of this Dictionary is available as follows:

- Hypertext Webster Interface
  Webster's Dictionary online with hypertext connections to related words within each definition (Size 4.1K):

  http://c.gp.cs.cmu.edu:5103/prog/webster

- Merriam-Webster WWWebster Dictionary
  [Help] | [New Search] Type in your word or phrase and press ENTER/RETURN. © 1996, Merriam-Webster, Inc (Size 1.5K):

  http://www.m-w.com/netdict.htm

2.34 System Items

Use the construction "the *name* item" when identifying an existing system item such as a command, function, parameter, and so on.

Reverse the construction when identifying a hypothetical system item or something the user must create; for example, "the file **/etc/OLDpasswd**."

Do not rely solely on typographic conventions to identify a system item.

Capitalize system item names as they appear on the system. All system items which are case-sensitive must be presented correctly.

Do not place quotation marks around system item names.

Do not hyphenate variable names used with commands.

Use system item names only as nouns or adjectives; do not use them as verbs.

Avoid starting sentences with the name of a system item or other name that begins with a lowercase letter. However, if you must begin a sentence with such a name, do not change the way it is capitalized.

The following system items should be semantically identified in source files:

arguments
array names
bits
character classes
constant expressions
constants
data structure fields or members
data structure names
environment variables
error values
external variables
file names
flags
functions
global variables
group names
headers
keyboard legends
macros
modifiers
operands
options
parameters
return values
signals
symbolic limits
tokens
user-defined structure names
utilities

2.35 Tables

Tables should be kept as simple as possible to enable conversion to other formats and display on electronic media with display limitations (such as when using the *man* command on a dumb terminal).

Introduce tables with a complete sentence, followed by a colon when the sentence immediately precedes the table. For example, "The ASCII codes for these functions are shown in the following table:" If you cannot avoid intervening text between the lead-in sentence and the table, use a period at the end of both the lead-in sentence and the intervening text.

Capitalize the first word in each table cell, regardless of whether the cell contains a complete sentence (the only exception is when the word is a literal term that must begin with a lowercase letter). Do not capitalize any other word within a table cell, unless the word is a proper noun, begins a sentence, or is a literal term that must be capitalized.

Abbreviations can be used in table cells where space is limited, but they must be defined. Notes can be used in tables to define abbreviations. Use superscript numbers for notes.

When a common unit of measure applies to all entries in a column, abbreviate it or spell it enclosed in parentheses after the column head. When units of measure are not common to all entries in a column, include the appropriate unit of measure with each entry in the column.

Align columns of decimal numbers on the decimal point, adding trailing zeros for consistency if necessary.

Use parallel construction in table text.

Table titles should use the conventions described in Section 2.37.

2.36 Times

Write specific times using the abbreviations a.m. (ante meridian, morning) and p.m. (post meridian, afternoon). (There is no space after the first period.)

If a time could be misinterpreted (for example, when discussing noon or midnight), you can also add the 24-hour equivalent. For example, ‘‘1:00 p.m. (13:00).’’

Midnight is defined as 12:00 a.m. (00:00).

Noon is defined as 12:00 p.m. (12:00).

Use the full name of time zones rather than their mnemonic codes. For example, Central European Time, Eastern Standard Time.

Do not refer to the date of changes to or from Daylight Saving Time, since this differs depending on the country.

Avoid using the words o'clock, noon, or midnight.

2.37 Titles

In general, make sure all titles are a reasonable length; ideally less than 60 characters.

Titles are used to start elements (such as chapters and appendices) sections, and subsections, and to label figures, tables, and examples.

Titles should summarize content.

Use titles levels 1, 2, 3, and unnumbered. Use of level 4 is discouraged, and use of lower levels is disallowed.

Use initial capitals for the following words in titles:

- The first and last words (always)
- Nouns, verbs, pronouns, adjectives, and adverbs
- Prepositions that contain four or more letters
- The second word in a hyphenated-compound (except articles, coordinating conjunctions, and system items that are case-sensitive)
- Abbreviations, acronyms, mnemonics, and keywords that are normally written in uppercase or lowercase. (If used, remember to define them in the following text.)

Do not use initial capitals for the following words in titles:

- The word “to” in infinitives
- Articles (a, an, and the)
- Conjunctions (and, but, as, and because)
- Case-sensitive system items

Avoid using an article as the first word in a title.

Do not begin a title with a technical term that must begin with a lowercase letter, such as the name of a function or utility.

Do not use typographic conventions in titles.

When a section describes an action to be performed (as in an example or procedure), you can begin the title with a gerund (for example, “Copying a Directory”). Sections that are more descriptive can use a noun phrase in the title (for example, “Output Formats”).

2.38 Trademarks

Trademarks are names, symbols, or other devices that identify products that are legally restricted to the use of the owner or manufacturer.

Trademarks should be spelled out and capitalized correctly.

Trademarked terms are not marked within text.

All documents for publication, whether in printed or online form, should include a list of trademark acknowledgements in the front matter.

Arrange trademark acknowledgements in alphabetical order.

Use trademarked names only as proper adjectives. For example, the word “UNIX” must be followed by the word “system.” Trademarks can therefore never be possessive or pluralized.

Do not use a trademark as part of a hyphenated compound.

The registered trademark UNIX must not be used to refer to the broader range of UNIX operating systems and their derivatives offered by other companies. UNIX is *now* a brand applied to systems that are branded to the appropriate Open Group specification, and *not* to product implementations.

If an abbreviation or acronym is also a trademarked term, do not spell it out.

It is the responsibility of the author of the document to provide a list of all the trademarks mentioned, although these will also be verified by The Open Group.

2.39 Warnings

Use a warning to draw the reader’s attention to information which, if ignored, can have a critical impact. See also Section 2.28 on page 16 and Section 2.6 on page 8.

Chapter 3

Reference Pages

3.1 Introduction

A *reference manual* is a document with a highly structured design that makes it easy for users to find information on a particular component or topic. The basic unit of information in this documentation is a *reference page*. This guide uses the term reference page to identify an entry in a reference manual.

A reference page may document a single system item, a descriptive topic, a sample program, or some other system feature. This guide uses the term *item* to describe the system component that is being documented in a reference page, and the term *reference page name* for the title of the reference page on which the item is documented. (The reference page name is derived from the name for the item, but the two are not always identical.)

On UNIX systems, the system reference manual is always available online, and users can access the manual from the command line by using the *man* utility. (The name of this utility is an abbreviation of the word *manual*.) For this reason, each reference page ideally should be a separate file, with a name that can be associated with the item being described.

Try to use the name of the system element being documented as the filename. This name cannot include a slash (/), but can include underscores (_). For this reason, the name may differ from the actual system name of the item. If a suffix is part of the actual system name, include it in the filename.

For example, the reference page for the **<sys/time.h>** header is **systime.h**.

If source files are stored on systems that do not support long file names or more than one suffix on a file name, be careful not to create duplicate file names when file names are truncated. In this situation, you must also supply a clear mapping of abbreviated file names to full file names (and reference page names).

Each reference page corresponds to an entry for the *man* utility. (The terms *manual page*, *man page*, and *manpage* are all synonyms for the term *reference page*.) These terms make sense as long as you understand that *page* refers to the unit of reference information, and does not necessarily correspond to a page in a printed book.

Do not refer to the subject of a reference page as ‘‘this *name*’’. Generally, you should refer to the subject of a reference page by name at least the first time it appears in each paragraph; for example, ‘‘The *mknod*( ) function.’’

3.1.1 Divisions

Reference pages may be combined into collections of information, either for external publication or to manage information. For this reason, reference pages are classified into divisions.

A reference page belongs to one of the following divisions, depending on the system item that the reference page describes:

User Utilities
: Describes utilities that are available to all system users.

System Administration Utilities
: Describes utilities that are used to manage systems and networks. System and network administrators read reference pages in this division.

System Calls
: Describes system calls which are functions that are entry points into the operating system kernel. Application developers read reference pages in this division.

Library Routines
: Describes library routines which are functions or macros that are included in libraries. Application developers read reference pages in this division.

File Formats
: Describes formats of headers (include files), program input and output files, and some system files. Application developers and system and network administrators read reference pages in this division.

Special Files
: Describes device special files, related drivers, and networking support available on the system. Application developers and system and network administrators read reference pages in this division.

Miscellaneous and Descriptive Topics
: Provides descriptive information on miscellaneous topics, or describes macro packages used for text processing. All users read reference pages in this division. Introductory pages typically fall into this division.

Examples and Demos
: Describes online examples, sample programs, demos, and games available on the system. All users read reference pages in this division.

3.1.2 Shadow Pages

For a page that documents multiple system items, you should include shadow pages for each system item and external variable (if applicable).

Shadow pages are reference pages that do not have content of their own. Instead, a shadow page points to another page. For example, the reference page for the *write*( ) function also includes documentation for the *writev*( ) function. The name of that reference page is *write*; but you can create a shadow page for a reference page named *writev*.

Shadow files are also used to provide documentation for external variables that are documented on the reference page for a function or header.

3.2 Writing Style

The guidelines documented in Chapter 2 on page 5 should be followed for reference pages. However, this section documents exceptions and additions to those guidelines which are specific to reference pages.

3.2.1 Abbreviations, Acronyms, and Mnemonics

Abbreviations, acronyms, and mnemonics should be defined (using the full name with the short form in parentheses) the first time the term is used in *each* reference page.

3.2.2 Cross-References

Cross-references in reference pages should be limited to the following:

- Within a reference page, you can cross-refer to another section, or to an example, within the same reference page.
- You can refer to another reference page.

All references to other references pages and documents should be listed in the SEE ALSO section.

Avoid referring to a specific section in another reference page. If required, use a descriptive phrase instead (for example, ‘‘see the information on portable file-name characters in ...’’).

To refer to other sections within the same reference page, do not precede the section name with *the*; for example, ‘‘see EXTENDED DESCRIPTION’’.

3.2.3 Examples

Reference pages should include examples for both naive and experienced users.

Short examples can be included to illustrate specific points in the descriptive sections of a reference page. More complete examples can be included in the EXAMPLES section. In the EXAMPLES section, each example should have a title so that it can be referred to from elsewhere in the reference page.

Examples of programming interfaces in a reference page are typically fragments of programs, but separate reference pages containing complete sample programs can be used to show programming calls in context. These complete sample programs can be used as source for example fragments, or you can simply refer the reader to the sample program reference page for examples.

Complete program examples should include extensive comments as part of the program text.

Consider adding examples for the following situations:

- Typical uses of a function or command (if not obvious)
- Usage suggested in APPLICATION USAGE text
- Clarification of a point in the text
- Illustration of the differences between related functions or commands
- Complex usage of a function or command

3.2.4 External References

If you need to refer to an external document, use an abbreviated document title within the body of the reference page and include it in the SEE ALSO section. (As before, the full bibliographic details should be included in the Referenced Documents section.)

3.2.5 Graphics

Reference pages should not include graphics, because pictorial information cannot be included when the pages are displayed using the *man* utility.

3.2.6 Index

The minimum level of indexing for reference pages is a single primary index entry for the name of the item documented on the page.

For reference pages that describe more than one item, include an index entry for each one.

Further index entries may be added as required.

3.2.7 Tables

When creating tables for reference pages, keep in mind that they may need to be displayed in different ways, including on character displays without proportional fonts (as when the *man* command is used on a dumb terminal). For this reason, tables should be kept as simple as possible.

Tables in reference pages should not have captions.

3.2.8 Titles

First-level sections in reference pages have standard titles.

Standard headings that are worded as plurals—such as ERRORS or EXAMPLES—should not be changed to singular when there is only one item in the section.

Subheadings can be used within these standard first-level sections, when it is necessary to subdivide a section. If subheadings are necessary, try to give at least two headings at each level (this is not an absolute requirement for reference pages, but it is the preferred style).

3.3 Reference Page Sections

Reference page sections use standard names and are ordered in a standard way. Inclusion of the various sections is dependent on the document type, as shown in the following table:

Reference Page Section	Product Documentation*	Shadow Pages	Specifications	Shadow Pages
NAME	M	M	M	M
SYNOPSIS	M[1]	M	M	M
DESCRIPTION	M	M	M	M
SUBCOMMANDS	X	X	O	X
OPTIONS	M (U)	X	M(U)	X
OPERANDS	O (U)	X	O(U)	X
PARAMETERS	O (I)	X	O(I)	X
EXTENDED DESCRIPTION	O	X	O	X
EXIT STATUS	M (U)	X	M(U)	X
RETURN VALUES	M (I)	X	M(I)	X
ERRORS	M (I)	X	M(I)[2]	X
EXAMPLES	O	X	O	X
ENVIRONMENT VARIABLES	M[3]	X	M[3]	X
ASYNCHRONOUS EVENTS	X	X	O (U)	X
FILES	O	X	O	X
NOTES	X	X	O	X
CAUTIONS	X	X	O	X
WARNINGS	X	X	O	X
DIAGNOSTICS	X	X	O	X
APPLICATION USAGE	X	X	O	X
FUTURE DIRECTIONS	X	X	O	X
SEE ALSO	O	X	O	X
CHANGE HISTORY	X	X	M[4]	M[4]

M Mandatory
O Optional
X Exclude
U For utilities.
I For interfaces.
* Includes Common Product Documentation.
1 Except in descriptive pages.
2 This information may be included in RETURN VALUES.
3 If environment variables affect the item.
4 If the item was described in a previous Open Group document.

3.3.1 NAME

Lists the exact names of one or more items discussed in the reference page and provides a very brief description of what each item does.

If more than one item is documented on a reference page, the item that is listed first is used as the title of the reference page. In some cases (such as the *exec*( ) functions), a *group name* can be used to represent all of the items on the page.

The description for the page is a short phrase that describes the item documented on the page:

- For functions, macros, and utilities, begin the description with an imperative verb (for example, ‘‘list directory entries’’).
- For descriptive reference pages, use a noun phrase (such as ‘‘conformance’’).
- For headers, begin the description with the phrase ‘‘include definitions for ...’’.

- For sample programs, begin the description with ‘‘sample program for ...’’.

The description should include articles as needed for readability. For example, ‘‘compress a file’’ is preferable to ‘‘compress file.’’ Do not use end punctuation or tag individual words within the description.

Do not capitalize the first word of the description unless it is a proper noun or a literal term that must be capitalized.

3.3.2 SYNOPSIS

The SYNOPSIS summarizes the syntax for an item.

For utilities, this section shows the command-line syntax. It should provide complete reference information, including all options, option-arguments, and operands. To ensure that the SYNOPSIS is useful as a quick reference, avoid using general terms to represent a group of options or operands.

First, list the options that do not take arguments. Group required and optional options separately. Do not group options that are mutually-exclusive—instead, separate them using vertical bars.

List options in alphabetical order. When options are listed in separate groupings, list them in alphabetical order within each grouping.

For example:

```
utility-name [-AaBcXxyz] [-p option-argument]
```

Mutually exclusive forms of a utility, or distinct ways of using a utility, should each be shown.

For functions or macros, this section shows the include statement and the form used for program calls; it may also show external variable declarations. It should provide complete reference information, including all parameters.

If multiple interfaces are documented on the same reference page, each should have its own entry.

Put mandatory parameters before optional parameters, unless the item requires otherwise. Indicate parameters of indeterminate number with an ellipsis following the parameter name. Do not use plural parameter names.

For headers, this section shows the include statement for the header. It may also show external variable declarations.

For sample programs, this section shows the compile line for the program.

For descriptive reference pages, this section is omitted.

3.3.3 DESCRIPTION

The DESCRIPTION provides a summary description of the item being documented.

For a function, macro, or utility, this section explains how the item is used. The description should be detailed enough so the reader can understand the standard use of the item, but should be kept quite short. (More detailed information should be presented in the EXTENDED DESCRIPTION section.)

A good summary should provide the following information:

- Purpose of the item (in general terms).
- How to control the item to perform different tasks. In explaining different uses, you can mention specific options, operands, or parameters, but should not discuss them in depth. For a utility, you should also explain how the utility operates when no options are specified.
- Common applications of the item.
- Input and output for the item.

If more than one item is documented on the page, this section should give some idea of the differences among them.

Unless all of the descriptive information for the item can be contained within 8 to 10 lines of text, avoid explaining the internal operation of a function, macro, or utility; discussing differences among implementations; mentioning specialized uses; or presenting output formats. Such information should instead be included in the EXTENDED DESCRIPTION section.

This section should use the same names for option-arguments, operands, and parameters as those used in the SYNOPSIS section.

Reference pages for descriptive topics do not include EXTENDED DESCRIPTION sections. Instead, they use subheads within the DESCRIPTION section. The same approach is used for reference pages that document files and sample programs.

Shadow pages should use this section to cross-refer to another reference page.

3.3.4 SUBCOMMANDS

The SUBCOMMANDS section describes a utility's subcommands, if any.

3.3.5 OPTIONS

The OPTIONS section lists all the options for a utility and provides a description of each option. The options are described in a definition list that is introduced by a lead-in sentence. Begin each description with a verb in the present tense, beginning with a capital letter (such as ‘‘Specifies’’ or ‘‘Indicates’’).

Information that applies to all options should precede the list, so that readers will not overlook it.

List options in alphabetical order.

For each option, specify whether it is mandatory or optional, or is mutually-exclusive with another option.

If no options are included with a utility, use standard text to state this fact.

Do not start a sentence with an option; use ‘‘The −z option ...’’

Arguments that are listed in the SYNOPSIS section, but are not associated with a particular option, are *operands*. Operands should be described in the OPERANDS section.

3.3.6 OPERANDS

This section provides a list of the arguments that are supplied directly to a utility, rather than to one of its options. Operands generally follow the options on the command line.

The operands are described in a definition list that is introduced by a lead-in sentence. Begin each description with a verb in the present tense, beginning with a capital letter (such as "Specifies" or "Identifies").

List operands in alphabetical order.

3.3.7 PARAMETERS

This section describes the arguments supplied to a programming interface (system call or library routine).

The parameters are described in a definition list that is introduced by a lead-in sentence. Begin each description with a verb in the present tense, beginning with a capital letter (such as "Specifies" or, for a pointer, "Points to"). Do not include asterisks for the pointer, either on the term or within the definition.

List parameters in alphabetical order.

3.3.8 EXTENDED DESCRIPTION

This section gives more detailed information about the use of the item being documented, and about its behavior and features. It expands on subjects discussed in the DESCRIPTION section, but with more detail and background information. Use this section to discuss the internal operation of a function, macro, or utility; to point out differences among implementations; to mention specialized uses; and to present output formats.

This section should use the same names for option-arguments, operands, and parameters as those used in the SYNOPSIS section.

Implementation-dependent information within the EXTENDED DESCRIPTION section should be discussed at the end of a paragraph (see Chapter 5 on page 39).

The EXTENDED DESCRIPTION section may use short examples to clarify a point being made. If an extensive example is needed, include the example in the EXAMPLES section, and cross-refer to it.

Information within the EXTENDED DESCRIPTION section can be organized under subheadings if the information is extensive and their addition improves readability. As far as possible, you should use standard titles for these subheadings, such as Standard Input, Standard Output, Standard Error, Input Files, Output Files, Consequences of Errors.

3.3.9 EXIT STATUS

This section describes the values returned by a utility after completion. These values differ from those in the RETURN VALUES section, which are used for programming interfaces.

The exit status values are described in a definition list that is introduced by a lead-in sentence (use standard text for the lead-in sentence).

List the exit status for successful completion first. If appropriate (based on the source information), begin the description with the phrase "Successful completion." If additional information follows, use a colon (:) after the initial phrase, followed by a sentence beginning with a lowercase letter.

Next, present the exit codes for errors in numerical order. If appropriate (based on the source information), begin the description with the phrase ‘‘An error occurred.’’ If additional information follows, use a colon (:) after the initial phrase, followed by a sentence beginning with a lowercase letter.

3.3.10 RETURN VALUES

This section lists the returned values or codes for a programming interface (system call or library routine), followed by a description for each. (If appropriate, a description may say that a return value is unused or reserved for future use.)

Use a definition list for return values. (This may not be the best approach when the return values are not literals.) If more than one function or macro is included on a reference page, you may need to include more than one list of return values. Each list is introduced by a lead-in sentence that indicates the name of the functions to which each group of return values applies.

The return values for success and failure are clearly identified using standard phrases: ‘‘Success’’ or ‘‘Failure.’’ If additional explanation follows, the phrase is ended with a colon, and the explanation follows in a sentence that begins with a lowercase letter.

3.3.11 ERRORS

This section documents *errno* values set by a programming interface (system call or library routine).

Usually an interface returns a numeric value to indicate failure, and it also sets *errno* to specify the reason. The structure of this section allows quick access to the error information.

Use the definition list format for errors.

Errors should be categorized depending on whether they are required or optional for a conforming system. Use standard text for the lead-in sentences for each such grouping of errors.

If multiple interfaces are documented on the same reference page, the errors should be divided by interface (unless grouping can eliminate duplication without sacrificing clarity). If the *errno* values differ in the description of the condition, the *errno* values should be separated for each interface.

List error codes in alphabetical order within each grouping.

If no errors are defined, or if the specification indicates that only the errors listed in the specification can occur, include standard text to state that fact.

3.3.12 ASYNCHRONOUS EVENTS

This section lists how a utility reacts to such events as signals and which signals are caught.

3.3.13 EXAMPLES

This section should be included whenever possible, to show how functions or utilities can be used. Although tutorial information does not appear in reference pages, examples are often useful to help readers understand how to use a function or utility. Examples should be sufficiently complete real-world examples of actual user tasks, unlike the code fragment examples in the EXTENDED DESCRIPTION section, which simply clarify a point.

Do not manually number examples; this should be handled by the output format.

Command and utility examples should show how to use the utility to accomplish a specific task.

Interface examples should show how to use the interface in a program. The example need not be a complete, compilable program. It can be a code fragment. If the example is a complete program, the example should be tested to be certain it will compile and link on a target system. An introductory sentence should indicate that it is a compilable program.

Complete sample programs may be contained in separate reference pages, and can be used as a source of examples or referred to from other reference pages.

If the SYNOPSIS tells you everything you need to know about a utility or function, no examples are needed.

The title for an example should describe what the example is demonstrating. When creating a title for an example that shows how to use a utility or function to perform a particular task, use a gerund (that is, a verb ending in ‘‘-ing’’) to indicate that an action is occurring. In addition, the title should usually describe the task in general terms, identifying a particular technique or task, rather than simply listing the constructs that are used in the example. For example, an example for the *tar* utility might be entitled ‘‘Creating an Archive.’’ This title is preferable to one such as ‘‘Using the −c Option.’’

For examples that do not show how to perform specific tasks, such as a sample file or a list of expressions, use a noun form such as ‘‘Sample File Access Permissions.’’

3.3.14 ENVIRONMENT VARIABLES

This section lists and describes the environment variables that affect the system item. It appears in most reference pages for utilities, but is rarely used in reference pages for system calls or library routines.

Environment variables are always spelled using all uppercase letters.

The environment variables are described in a definition list that is introduced by a lead-in sentence. Begin each description with a verb in the present tense, beginning with a capital letter.

Environment variables sometimes have the same names as locale categories. Be careful not to confuse them. You do not need to list or describe locale categories in the ENVIRONMENT VARIABLES section.

In general text, where only a name (such as LC_CTYPE) is used in the source, you should add text to indicate whether an environment variable or a locale category is being discussed.

List environment variables in alphabetical order.

3.3.15 FILES

This section lists all of the files or directories of files, other than those having user-supplied names, that are read, created, modified, or otherwise touched by an interface or utility. List all such files in a definition list, and give a brief description of each file.

Include files such as system-supplied configuration, initialization, or log files in this section. Do not include user-supplied input files or output files.

List files in alphabetical order.

3.3.16 NOTES

This section includes any supplementary information that is peripheral to the actual operation of the system item. Use this section instead of individual notes in other sections.

3.3.17 CAUTIONS

This section includes information on possible system damage or data corruption that may occur as a result of using the system item in a specific implementation.

3.3.18 WARNINGS

This section includes information which, if ignored, can have a critical impact.

3.3.19 DIAGNOSTICS

This section provides information useful for diagnosing errors that may result when the system item is used.

3.3.20 APPLICATION USAGE

This section includes information by which the user can avoid misuse of the system item.

3.3.21 FUTURE DIRECTIONS

Include this section if there are firm plans to change the item in the future or to introduce new features that address a specific limitation, or if there is a known uncertainty which is expected to be resolved.

3.3.22 SEE ALSO

This section lists other reference pages that are referred to in the reference page, along with any others that are relevant. References to standards documentation may be made here, in a separate paragraph.

The list of reference pages is grouped by reference manual divisions, and alphabetized within each group (ignoring case). The groups are listed in the following order:

- Utilities (user command and system administration divisions, ordered as a single division)
- Programming Interfaces (system call and library routine divisions, ordered as a single division)
- File Format
- Miscellaneous (descriptive pages)

- Device
- Example

Where referring to a system item that is not the main item on another reference page, use the main reference page name.

After listing the reference pages, you can list any other document references that are appropriate in a separate paragraph, also in alphabetical order.

Do not include the word *and* before the last entry, or type a period after the last entry, in either of these paragraphs. Separate the entries with a comma.

3.3.23 CHANGE HISTORY

This section indicates the first document in which the system item appeared, and briefly summarizes all changes applied since the previous issue, if any.

Chapter 4

Product Documentation

4.1 Introduction

Documentation plays an important part in how well a product is received by its users.

Most readers approach new product documentation with similar attitudes:

- They are eager for action.
- They are primed to do work with the product.
- They are short of time.
- They are motivated to succeed.

Readers can learn more quickly if documents are written for action. Where appropriate, introduce simple tasks or procedures very early, and make sure users are certain of success in performing those procedures.

Research shows that readers approach learning about new products in two ways, with two different learning styles.

Experimenters want to try things right away instead of taking the time to read the instructions. Experimenters share these characteristics:

- They read written procedures only as a last resort.
- They believe instructions are probably incomplete.
- They rely heavily on tables of contents, section titles, and indexes when searching for information.
- They consult examples more often than text for information.

Directed learners read before they act. Directed learners share these characteristics:

- They read procedures and explanations carefully.
- They believe the documentation is correct.
- They pause over tiny discrepancies.
- They consult examples to confirm the correctness of their actions.

Most documentation should provide information that works for both the experimenter and the directed learner. Where appropriate, include step-by-step teaching exercises (tutorials) for directed learners and easily accessible reference and procedural information for experimenters.

Experimenters make heavy use of the table of contents, section titles, and the index to find information. Directed learners often read straight through a document and later return to the same elements for review.

A comprehensive index is the most useful aid you can provide a searching reader; a clear table of contents is almost as important. When you write section titles, make them lively and direct so that readers can tell exactly what a section is about. The table of contents then becomes much more useful as a pointer to information.

Your knowledge of the product—its uses, benefits, and idiosyncrasies—is critical to developing useful product documentation. Occasionally, writers try to develop documentation based solely on the developers' specifications. Even when the developers are very capable writers, the documentation suffers.

Get your hands on the product and use it as much as time allows. When working with the product, you learn to use it effectively, and you can pass along to readers many tips and shortcuts you could not otherwise have provided.

As you think about potential users of your documentation, keep in mind the product's market—what kind of individuals or businesses will use it, how they will use it, and what benefits the product can provide. Consider the internationalization and localization issues that must be addressed as you plan the development of your document. Questions like the following can be useful when examining potential users of your document:

- Who are the users?
- What does the product do for the users?
- What are the product's benefits?
- Where will the product be used?

4.2 Document Structure

Product documentation includes the following:

- User Documentation

  Documentation that describes how to install, operate, and manage software.

- Development Documentation

  Documentation that describes how to develop, modify, and port software to specific platforms. Examples are coding style guides, programmer's reference manuals, and software porting guides.

The following table shows the components that are required in all product documentation and the order in which they appear. Most components are required in all volumes of a multi-volume documentation set.

Component	First or Only Volume	Other Volume
Title Page	M	M
Copyright Page	M	M
Restrictions	M	M
Warranties*	R	R
Acknowledgements	M	M
Contents	M	M
List of Illustrations	M	M
List of Tables	M	M
Preface	M	R
Audience	M	M
Applicability**	M	M
Purpose	M	R
Document Usage	M	R
Related Documents	R[1]	R
Typographic and Keying Conventions	M	M
Problem Reporting	M	R
Document Body	X	X
Appendixes	O	O
Glossary	M	M[2]
Index	M	M[2]

Notes:

M Must be included when the information exists
O Optional.
R Reference; include within the volume or reference another document that contains the information.
X Dependent on document type.
1. Addresses relationship to other volumes.
2. Must be included in at least one volume; use references to the component in other volumes.
* Warranty information varies by country, and is often legally required in the local language. Warranty information should therefore be included in an addendum.
** Applicability is a new section which is added specifically for Product Documentation. It identifies the software product (including version number) to which the document applies. It includes, if relevant, information about the software and hardware environment required to use this software.

Chapter 5

Common Product Documentation

5.1 Introduction

Common product documentation is based on approved Open Group specifications and is designed to:

- Include all pertinent information in a way that is understandable and meaningful to all users
- Be consistent in organization and style
- Provide placeholders for integration of vendor-specific information

The inclusion of vendor-specific information enables the clear distinction between that which applies to all conforming systems and that which is vendor-specific. It also ensures that vendor-specific information is included in a way that is consistent across all vendor systems.

Common product documentation presents information in a form that is suitable for users, programmers, and system administrators. It provides clear explanations of system features, examples of their use, and complete reference information. It also distinguishes between common features that are available across all conforming systems and extensions that may be supported only on systems from a particular system vendor.

Language in specifications—because it often has a special meaning, and because it is addressed to system implementors—often needs to be revised for product documentation.

Specifications also include conformance information for branding purposes. This information is not necessarily used in product documentation, but the information should be retained in the document source.

This chapter documents variations from the writing style already described which are required for common product documentation.

5.2 Language

The language used in specifications is for implementors. As such, it often uses precisely defined terms that will be unclear to users of a system. To create common documentation it is often necessary to revise text to clarify the meaning for a reader who is more interested in how a particular system behaves, or how much they can rely on a particular behavior across systems that conform to the specification, than in how the specification defines a particular behavior.

When features or behavior may vary among different systems, you should add a *vendor placeholder* that can be used to present system-specific information, so that system vendors can document the differences on their systems. In addition, when you create a vendor placeholder, you should include a comment to the vendor (which does not appear in formatted text) indicating the information that should be added. Information that is optional for vendors to add should be marked with the phrase ‘‘VENDOR ADDITION:’’ within the comment. Information that vendors must add for conformance to the specification should be marked with the phrase "VENDOR REQUIREMENT:" within the comment.

In addition, specifications often abbreviate the information for some items, by referring to other volumes of the specifications, by referring to a glossary entry, or by specifying the behavior of a system item in relation to another system item, instead of spelling out exactly how it works from a user's perspective.

The audience for common product documentation is interested in a clear answer to the following questions:

- Will all systems that conform to the specification include this feature or behavior? If not, reword and add a vendor placeholder.
- More specifically, if I am writing an application or script that must be portable, can my script or application depend on this feature or behavior? If not, reword and add a vendor placeholder.

When vendor-specific information is integrated, users also need a clear answer to the following question:

- What are the specifics for the product I am using now? Add a vendor placeholder when the specification does not define a particular feature or behavior, so that vendors can explain how it works on their systems.

The reader must also be able to distinguish clearly between base information (which applies to all conforming systems) and vendor-specific information that might be added in a vendor placeholder (which applies only to a specific system). Further, the presentation must make sense both when vendor-specific information is omitted (because vendors are not required to add information for every vendor placeholder) and when vendor-specific information is added.

5.3 Terminology

The following terms are indicators that you may need to add a placeholder for vendor-specific material, or otherwise adapt text from the specification:

- Can
- Implementation-dependent

  In a specification, this term refers to values or behavior not defined by the specification. Users are likely to need, and therefore vendors should supply, implementation-dependent information in order to write and debug programs or scripts. The source code should supply a vendor placeholder.

  The term "implementation-dependent" can be ambiguous when it is used in product documentation, because the term can refer to variations among a vendor's product lines, as well as to each vendor's implementation of a feature according to a standard. Therefore, reword the sentence to explicitly note any portability implications and to ensure graceful integration of product information by vendor writers. For example:

  In a specification:

  > The behavior of *fseek*( ) on devices that are incapable of seeking is implementation-dependent.

  In common documentation:

> When the *fseek*( ) function is used for devices that are incapable of seeking, the results may vary among systems that conform to the specification.
>
> [vendor placeholder]

The following example shows a case in which the specification requires that vendor-specific information be added:

In a specification:

> An implementation will document any condition not specified by this document under which the implementation generates signals.

In common documentation:

> VENDOR REQUIREMENT: The specification requires a system vendor to document any conditions that cause signal generation and that are not described in the specification. The following placeholder is provided to enable vendors to add any appropriate conditions.
>
> [vendor placeholder]

In this example, no text actually appears in the formatted document unless the vendor adds information for the placeholder; but the information is required for conformance to the specification.

- May

  The term ‘‘may’’ indicates optional behavior that may vary among conforming systems, so you should state that point directly and add a vendor placeholder:

  Because the term ‘‘may’’ implies variation among different systems, you should include a vendor placeholder when this word is used in a SYNOPSIS, DESCRIPTION, RETURN VALUES, or ERRORS section.

  For example, in a specification:

  > This function may return an error code of −1 if an error occurs.

  is changed to:

  > On some conforming systems, this function returns an error code of −1 if an error occurs.

  In an Errors section:

  > The *fopen*( ) function may fail if: [list of errnos and conditions]

  is changed to:

  > The *fopen*( ) function can set *errno* to one of the following if the corresponding error condition occurs, but systems that conform to the specification are not required to detect these conditions:
  >
  > [list of errnos in the specification’s ‘‘may’’ list and vendor placeholder]

- Obsolescent

  Some text in the specification may be marked as obsolescent using the marginal tag OB, with shading of the text that applies to the obsolescent feature. Remove these markings.

  In addition, you should add text such as the following at the end of the DESCRIPTION section:

The *grep* −**F** utility is the recommended alternative to the *fgrep* utility, which may be withdrawn in a future version of the specification.

- Should
- To be withdrawn

Reword this terminology to avoid misunderstanding.

For example, in a specification:

cc — a C-language compilation system (TO BE WITHDRAWN)

is changed to:

cc - a C-language compilation system

In addition, add text such as the following at the end of the DESCRIPTION section:

The *cc* utility is scheduled to be withdrawn from a future version of the specification. The *c89* utility is the recommended replacement.

- Undefined

You should spell out what these words mean because vendor writers may have to describe error detection and behaviors that the specification says are undefined and unspecified. It is confusing for a reader to read that something is unspecified in one section of a reference page and then read the behavior specification somewhere else. Mark-up techniques do not entirely eliminate this confusion, especially when readers are searching the reference page to find specific strings.

Here are some examples of how to reword this terminology in reference pages.

In a specification:

It is unspecified whether writes to the same portion of the file prior to the *msync*( ) call are visible by read references to the memory region.

In common documentation:

On some systems that conform to the Single UNIX Specification, read references to a specific memory region may not be able to access the results of write operations that were made to the same portion of the file before the current *msync*( ) call.

This rewording allows the vendor writer to incorporate, if necessary, a paragraph like the following one without altering the text supplied by the specification:

On *name* systems, read references can access write operations made to the file prior to the call.

In a specification:

If an insufficient number of arguments is passed to accept all the values that the regular expression returns, the behavior is undefined.

In common documentation:

If the call does not pass a sufficient number of arguments to accept all the values that the regular expression returns, the results may vary among systems that conform to the Single UNIX Specification.

[vendor placeholder]

- Unspecified

  Refer to Undefined.

- Will

  When the term "will" is used in a SYNOPSIS, DESCRIPTION, RETURN VALUES, or ERRORS section, you can reword the sentence to simply describe the feature or behavior as the way the item works (on all systems that conform to the specification).

  Depending on context, the verb can be eliminated entirely, or may need different wording. For example, in the following text the word *will* indicates the required behavior of the function, so the sentence can be rephrased to use the present tense:

  In a specification:

  > This function will return an error code of −1 if an error occurs.

  In common documentation:

  > This function returns an error code of −1 if an error occurs.

  The following examples shows the use of "will" in an ERRORS section:

  In a specification:

  > The *fopen*( ) function will fail if: [list of errnos and conditions]

  In common documentation:

  > On all systems that conform to the specification, the *fopen*( ) function sets *errno* as listed for the following conditions: [list of the errnos in the specification's "will" list]

5.4 Equations

Equations should be followed by a vendor placeholder that can be used to include an alternate version of the equation.

5.5 Reference Pages

Copyright

Each reference page should contain a copyright notice as part of the document source. This copyright statement, added as a customizable text entity, assigns the copyright to The Open Group. When vendors integrate reference pages into their own product documentation, they should add their own copyright statements to the file as a separate line.

SYNOPSIS

Do not change the order of parameters in a function synopsis; use the same sequence as in the specifications.

OPTIONS

System vendors can define additional options for a utility, so you should include vendor placeholders before and after the option list. The placeholder before the option list can be used to indicate that additional options are defined. The placeholder following the list can be used to list the additional options.

OPERANDS

System vendors can define additional operands for a utility, so you should include vendor placeholders before and after the operand list. The placeholder before the operand list can be used to indicate that additional operands are defined. The placeholder following the list can be used to list the additional operands.

EXTENDED DESCRIPTION

Implementation-dependent information within the EXTENDED DESCRIPTION section should be discussed at the end of a paragraph, so that system vendors can add information following the paragraph, in a modular fashion.

There are several specification section headings that should not appear as section headings in common product documentation reference pages. They are:

- STDIN (Standard Input)
- STDOUT (Standard Output)
- STDERR (Standard Error)
- INPUT FILES
- OUTPUT FILES
- CONSEQUENCES OF ERRORS

The information contained in these sections should be added to the reference page EXTENDED DESCRIPTION section. Do not use these section headings in the reference pages unless the information they contain is extensive or significant, and their addition improves readability. In this case, use the specification's main headings as subheadings in the EXTENDED DESCRIPTION section.

If they are used as headings, spell out the headings Standard Input, Standard Output, and Standard Error.

You may also include nonstandard subheadings (such as ''Output Format'' and ''Interactions Among Options'') if their inclusion makes the subject matter easier to scan.

In reference pages that document multiple programming interfaces, the information for each interface should be kept as a block of text that is identified by a subheading (unless the information is very brief). The subheading should use the interface name as the title of the section.

Some specification section headings contain information that may be included in the EXTENDED DESCRIPTION section. Include the information, but do not use these titles as subsection headings. They are:

- APPLICATION USAGE
- ASYNCHRONOUS EVENTS

Some specification section headings contain information that should not be included in the reference page. Do not include information from those sections. They are:

- CHANGE HISTORY
- FUTURE DIRECTIONS

RETURN VALUES

System vendors can define additional return values for a function or macro, so you should include vendor placeholders before and after the list of return values. The placeholder before the list can be used to indicate that additional return values are defined; the placeholder following the list can be used to list the additional return values.

ERRORS

In most cases, system vendors can define additional errors for a function or macro, so you should include vendor placeholders before and after each list of errors. The placeholder before the list can be used to indicate that additional errors are defined; the placeholder following the list can be used to list the additional errors.

EXAMPLES

One of the primary jobs of a reference page writer is to consider whether additional examples are needed for a particular reference page. If the specifications do not include examples, or if the examples are not helpful for the user of a command or for a programmer using a function, you should consider adding examples.

FUTURE DIRECTIONS

Not included.

CHANGE HISTORY

Not included.

Chapter 6

Presentation

The mechanism used for formatting in each mark-up language and production method is different. This chapter therefore describes overall preferred presentation style. It is designed as an aid only and is not an exhaustive list of instructions.

If more information is required, you should consult The Open Group rather than inventing individual presentation styles.

6.1 Contents

The Contents list should include entries to the third level of the document hierarchy, a page number for each entry, and leader strings connecting entries with page numbers.

There should be a list of Figures showing the number and title of each entry, a page number for each entry, and leader strings connecting entries with page numbers.

There should be a list of Tables showing the number and title of each entry, a page number for each entry, and leader strings connecting entries with page numbers.

6.2 Cross-References and External References

Initial capitals are used for cross-references.

References within a document give the section, figure, or table number, and page number.

If the item referred to is on the same page as the reference, no page number is given. If the item referred to is on a different page, the identification is followed by a string "on page *n*". (The page number may be suppressed.)

Use italic font for the titles of other referenced documents.

6.3 Drafts

Drafts should have line numbers and change bars (if required).

Identification of a draft should appear in the footer (see Section 6.13 on page 49).

6.4 Examples

Most examples appear in constant width font. User input may be identified using bold font. Variable information may be identified using italic font.

6.5 Figures

Figures should be centered and should not be surrounded by a frame or box.

Use the same typeface as the text, unless typographic and keying conventions apply.

The numbering scheme for figures consists of the chapter number followed by the figure number, separated by a hyphen. This text should be centered and placed below the figure.

6.6 Fonts

Most Open Group technical documentation is produced in Palatino Roman, font size 11 (on 13). Some technical-related documentation is produced in Helvetica, also font size 11 (on 13).

Punctuation is presented in the base font, unless it is part of syntax. See also Section 6.18 on page 53.

6.7 Footnotes

Use superscript numbers to refer to a footnote in text. Place the number after the key word in the text or table, and use the same number with the footnote itself.

A dagger or other symbol may be used where the same footnote applies to several points.

Place footnotes at the bottom of the page.

6.8 Glossary

Glossaries are formatted as a definition (or variable) list, with a short indent (4n) and a line break between the glossary term and the definition text.

6.9 Headings

Use initial capitals for the following words:

Chapter, Appendix, Section, Version, Release, Guideline, Table, Figure

The left margin should be sufficient for all heading (section) numbers.

The fonts and point sizes of the various heading levels are determined by the processing tools in use.

6.10 Index

Includes a list of indexed terms, in 2 columns, with a page number for each entry, and leader strings connecting entries with page numbers. Main index entries should make the page number appear in bold.

6.11 Lists

For unnumbered lists, use bullets for a first-level list, and em-dashes (—) for a second-level list.

For numbered lists, use Arabic numbers (1. 2. 3.) for a first-level list, and Letters (a. b. c.) for a second-level list.

6.12 Notes

Notes are highlighted by placing the text "Note:" in bold font and indenting the note text itself.

In drafts, editor's notes are identified with marginal shading and the text "Notes to Reviewers."

6.13 Page Layout

Most hard copy Open Group technical documents are published in AQ (8.5 x 11 inches), although the print area should accommodate A4.

All document elements begin on a right-hand, odd-numbered page, with the exception of the front matter sections which are run on.

The title page should include the series title, document title, and "The Open Group".

Page numbering is placed at the outer edge of the footer. Front matter is numbered using Roman numerals; the document body is numbered using Arabic numerals.

Page headers are as follows:

- The Chapter heading should be used as the running header on the spine side of the document.
- The current second-level heading should be used as the running header on the outer edge of the page.

Note there is no page header on Page 1 of a document element.

Page footers are as follows:

- The document title (or current part title, if applicable) should be used in the footer of every odd-numbered page.
- The series title and the year in parentheses should be used in the footer of every even-numbered page.
- For drafts, add ‘‘Draft’’ and a specific date to the even-numbered pages.

6.14 Pagination

Some required space should be built into each section or subsection heading, so that they cannot start near the bottom of the page.

To avoid widows and orphans, there should be at least two lines on each page.

Second-level headings typically start on a new page, except where the effect is to leave large amounts of white space that appears unnecessary. If the chapter is short (say, 10 pages or less in run-on form), all sections may be run on.

6.15 Reference Pages

Each reference page starts on a new page.

Reference page headings are displayed on the outer top edge of the page, as follows:

- For functions, in bold font with parentheses; for example, **function**( ). (Italic font is used in the Contents.)
- For COBOL functions, in bold font; for example, **function**. (Roman font is used in the Contents.)
- For headers, in bold font with angle brackets; for example, **<header.h>**. (Bold font is used in the Contents.)
- For macros, in bold font; for example, **macro**. (Italic font is used in the Contents.)
- For data types, in bold font; for example, **data_type**. (Bold font is used in the Contents.)
- For X Windows widgets, in bold font; for example, **widget**. (Italic font is used in the Contents.)

Reference page sections are arranged as a definition list with the section heading as the list term.

The NAME section is the name of the item described, followed by a space, followed by an em-dash, followed by another space, followed by the brief description.

The SYNOPSIS section should be in constant width with adjusting and filling disabled.

In utility synopses, options should be enclosed in brackets ([]), a group is enclosed in braces ({}), and a repeatable value is followed by an ellipsis (...).

In function and macro synopses, parentheses begin and end the parameter list, commas separate the parameter definitions, and a semicolon ends the function statement.

The other reference page sections follow standard text conventions as already described in this document.

The following example shows a typical reference page.

catclose( )

NAME

catclose — close a message catalog descriptor

SYNOPSIS

EX

```
#include <nl_types.h>

int catclose(nl_catd catd);
```

DESCRIPTION

The *catclose*( ) function closes the message catalog identified by *catd*. If a file descriptor is used to implement the type **nl_catd**, that file descriptor will be closed.

RETURN VALUES

Upon successful completion, *catclose*( ) returns 0. Otherwise, −1 is returned, and *errno* is set to indicate the error.

ERRORS

The *catclose*( ) function may fail if:

[EBADF]	The catalog descriptor is not valid.
[EINTR]	The *catclose*( ) function was interrupted by a signal.

EXAMPLES

None.

APPLICATION USAGE

None.

FUTURE DIRECTIONS

None.

SEE ALSO

catgets( ), *catopen*( ), **<nl_types.h>**

CHANGE HISTORY

First released in Issue 2.

Issue 4

The following change is incorporated in this issue:

- The [EBADF] and [EINTR] errors are added to the ERRORS section.

6.16 Shading

Shading is the preferred method of identifying extensions and warnings. A marginal mark indicates the type of extension or warning.

6.17 Tables

Tables should be centered and boxed.

Use the same typeface as the text, unless typographic and keying conventions apply.

Column headings should be centered and appear in bold. There should be a horizontal rule underneath them. Repeat the column headings when a table is continued onto one or more pages.

Use vertical rules to separate all of the columns.

Text in table cells should be flush left. However, if entries are 3 characters or less, they should be centered. Also, numerics may be aligned using the decimal point.

Use em-dashes (—) in cells that do not have information in them.

Here is a typical table:

Column 1	**Column 2**	**Column 3**	**Column 4**
1.0	1.5	2.0	2.5
2.0	2.5	3.0	3.5
3.0	4.0	5.0	6.0

The numbering scheme for tables consists of the chapter number followed by the table number, separated by a hyphen. This text should be centered and placed above the table.

6.18 Typograhical Conventions

The following typographical conventions are preferred:

- **Bold** font is used in text for options to commands, filenames, keywords, type names, data structures and their members.
- *Italic* strings are used for emphasis, to identify the first instance of a word requiring definition, or for foreign phrases. Italics in text also denote:
  - — Command operands, command option-arguments, or variable names; for example, substitutable argument prototypes
  - — Environment variables, which are also shown in capitals
  - — Utility names
  - — Macro names
  - — External variables, such as *errno*
  - — Functions; these are shown as follows: *name*( ); names without parentheses are C external variables, C function family names, utility names, command operands, or command option-arguments.

- Normal font is used for the names of constants and literals.
- The notation **<file.h>** indicates a header.
- Names surrounded by braces, for example, {ARG_MAX}, represent symbolic limits or configuration values which may be declared in appropriate headers by means of the C **#define** construct.
- The notation [EABCD] is used to identify an error value EABCD.
- Syntax, code examples, and user input in interactive examples are shown in `fixed width` font. Brackets shown in this font, `[ ]`, are part of the syntax and do *not* indicate optional items. In syntax the | symbol is used to separate alternatives, and ellipses (`...`) are used to show that additional arguments are optional.
- **`Bold fixed width`** font can be used to identify brackets that surround optional items in syntax, **`[ ]`**, and to identify system output in interactive examples.
- Variables within syntax statements may be shown in *`italic fixed width font`*.
- Ranges of values are indicated with parentheses or brackets as follows:
  - — (a,b) means the range of all values from a to b, including neither a nor b
  - — [a,b] means the range of all values from a to b, including a and b
  - — [a,b) means the range of all values from a to b, including a, but not b
  - — (a,b] means the range of all values from a to b, including b, but not a.
- Helvetica is used for the Pseudo Interface Definition Language (PIDL).
- **Helvetica Bold** is used for the Interface Definition Language (IDL).

6.19 Underlining

Underling is not used, except where it is required to conform with a referenced document, for example:

- COBOL syntax
- An extract reproduced from another document where underlining is used.

Appendix A

Terminology

This appendix gives selected terms, their meaning, and preferred spelling and alternatives, if applicable.

Open Group documentation is likely to be translated. Remember to convey important distinctions by using the right term.

ACL
Access Control List.

AES
Application Environment Specification (historical use only).

ANSI
American National Standards Institute.

API
Application Program Interface.

ASCII
American National Standard Code for Information Interchange.

above
Do not use to refer to another location in a document.

alphanumeric
When used as a modifier.

and/or
Avoid using this term. Rewrite the sentence, or use the two options, followed by "or both."

appendixes
Do not use *appendices*.

argument
Value provided to a function or utility (such as the value of a parameter, operand, or option-argument).

async
Use *asynchronous*.

B
Byte.

Boolean
Note capitalization.

BSD
Berkeley Software Distribution.

b
Bit.

back up
When used as a verb.

backslash
When used as a modifier or noun.

backspace
When used as a modifier, noun, or verb.

backup
When used as a modifier or noun.

baud rate
Often incorrectly assumed to indicate the number of bits per second (bps) transmitted. Baud rate actually measures the number of events, or signal changes, that occur in one second. In most instances when baud rate is used, the correct term is "bps." Check your source material before using the term baud rate.

behavior
Do not use "behaviour."

below
Do not use to refer to another location in a document.

big-endian
When used as a modifier.

boot
Start up a system.

built-in
When used as a modifier.

built-in utility
Special utility implemented within the shell. Also see *shell*.

C language
When used as a noun.

C-language
When used as a modifier.

CD-ROM
Note capitalization and use of hyphen.

COBOL
Note capitalization.

can
The term *can* describes a permissible optional feature or behavior available to the user or application. The feature or behavior is mandatory for an implementation that conforms to the specification. An application can rely on the existence of the feature or behavior.

cannot
One word.

cf
Expand to *compare*.

choose
Use this verb when picking an operation from a menu.

circa
Expand to *about*.

client-server
When used as a modifier.

codeset
When used as a modifier or noun.

commands
Call to the shell to perform a specific task. A string entered on the command line or in a script. Contrast with *utility* which is the name of an executable.

command-line
When used as a modifier.

cross-reference
When used as a noun.

DBMS
Database management system.

DCE
Data communications equipment.

DCE
Distributed Computing Environment.

DES
Data Encryption Standard.

DIF
Data interchange format.

DSR
Data set ready.

DTD
Document Type Definition.

DTE
Data terminal equipment.

DTR
Data terminal ready.

data
Use as a singular noun. Do not use *datum*.

data type
When used as a noun.

database
When used as a modifier or noun.

default
Value or behavior provided by the system.

dial up
When used as a verb.

dial-up
When used as a modifier.

directory name
When used as a noun.

disk
Use for any disk other than an optical disc.

diskette
State the size (3.5 or 5.25 inches), and do not use the modifier *floppy*.

display
Use the verb *display* rather than *appear*. For example, "The prompt is displayed on the screen." Avoid using *displays* without an object. For example, "The system response displays on the screen." (Use of the noun *appearance* is acceptable.)

dump file
When used as a noun.

EBCDIC
Extended Binary-coded Decimal Interchange Code.

EIA
Electronics Industry Association.

EISA
Extended Industry Standard Architecture.

EOF
End of file.

EOT
End of tape.

EOT
End of transmission.

Ethernet
Note capitalization.

e.g.
Expand to "for example," (note that addition of a comma).

et al
Expand to "and others."

email
When used as a modifier or noun.

end user
Person using a system feature.

end-user
When used as a modifier.

enter
To submit input to the system. Also see *type*.

Do not use *enter* to indicate the startup of an application.

entry level
When used as a noun.

etc.
Expand to ‘‘and so forth’’.

execute
To run a command using the current execution environment. Also see *invoke*.

FIFO
First in, first out.

FILO
First in, last out.

FIPS
Federal Information Processing Standard.

FORTRAN
Formula translation.

FTP
File Transfer Protocol.

fewer
Use to refer to countable items; for example, ‘‘you will find fewer errors’’.

file name
When used as a noun.

filename
When used as a variable in syntax or examples.

file sharing
When used as a noun.

file-sharing
When used as a modifier.

file system
When used as a noun.

fixed length
When used as a noun.

floating point
When used as a noun.

floating-point
When used as a modifier.

G
Giga (prefix).

Gbyte, GB
Gigabyte.

GID
Group identification.

general-purpose
When used as a modifier.

hard copy
When used as a noun.

hard-copy
When used as a modifier.

hexadecimal
When used as a modifier.

IEEE
Institute of Electrical and Electronics Engineers.

I/O
Input/output.

IRQ
Interrupt request.

ISA
Industry Standard Architecture.

ISA
Instruction-set architecture.

ISDN
Integrated services digital network.

ISO
International Organization for Standardization.

i.e.
Expand to ‘‘that is,’’ (note that addition of a comma).

if
Use when an event is conditional.

implementation
Refers to the way a function or utility works on a particular operating system.

implementation-dependent
(Same meaning as *implementation-defined*.) Describes a value or behavior that is not defined by the definitions contained in the specification but is selected by an implementor. The value or behavior may vary among implementations that conform to the definitions contained in the specification. An application should not rely on the existence of the value or behavior. An application that relies on such a value or behavior is not portable across conforming implementations.

The implementor normally documents such a value or behavior so that it can be used correctly by an application.

invoke
Run a command with suppression of searching for shell functions and special built-in utilities.

initialize
Not ‘‘initialise’’.

in-line
When used as a modifier.

input
Use as a noun only, not as a verb.

interconnect
When used as a modifier, noun, or verb.

interface
When used as a modifier or noun, not as a verb.

interprocess
When used as a modifier or noun.

Kb
Kilobit.

Kbyte, KB, or K
Kilobyte.

Korn shell
Not *KornShell*.

k
Kilo (prefix).

keyboard
When used as a modifier or noun.

LP
Line printer.

left-justified
When used as a modifier.

legacy
Describes a feature or behavior that is being retained for compatibility with older applications, but which has limitations which make it inappropriate for developing portable applications. New applications should use alternative means of obtaining equivalent functionality.

less
Use to refer to non-countable items or when discussing something in terms of size or degree; for example, ‘‘this is less complicated’’.

log file
When used as a noun.

log in
When used as a verb.

login
When used as a modifier or noun.

log out
When used as a verb.

logout
When used as a modifier or noun.

long-term
When used as a modifier or noun.

lowercase
When used as a modifier or noun.

MAC
Medium access control.

MAC
Memory access controller.

Mb
Megabit.

Mbps
One million bits per second.

Mbyte, MB
Megabyte.

MSB
Most significant bit.

mailbox
When used as a noun.

may
Describes a feature or behavior that is optional for an implementation that conforms to the definitions contained in the specification. An application should not rely on the existence of the feature or behavior. An application that relies on such a feature or behavior is not portable across conforming implementations.

To avoid ambiguity, the opposite of *may* is expressed as *need not*, instead of *may not*.

media
Use as a singular noun.

modem
Modulator.

mount
Make available to the system.

mouse
Use to refer to any pointing device, screen button, or menu operation. (Remember to define this usage.)

mouse button
Use to refer to a button on a mouse. Avoid the generic *button*. Use the verbs *click*, *double-click*, *drag*, *press*, *hold*, and *release* to refer to a mouse button.

ms
Millisecond.

multiplexer
When used as a modifier or noun.

multitasking
When used as a modifier.

multiuser
When used as a modifier.

must
Same meaning as *shall*; *shall* is the preferred term.

NaN
Not a number.

NFS
Network File System.

need not
The negative of *may*. Used in preference to *may not* to avoid ambiguity.

newline
When used as a modifier or noun.

nonzero
When used as a modifier or noun.

OR
Do not use as a verb. For example, instead of saying ‘‘OR-ing the bits’’ say ‘‘a logical bitwise OR of the bits’’.

OS
Operating system.

OSI
Open Systems Interconnection.

obsolescent
Describes a feature that may be considered for withdrawal in a future version of the specification. Such features are retained because of their widespread use, but are not recommended for new applications. Vendors may continue to support such features, even after they are withdrawn from the standard.

offline
When used as a modifier.

online
When used as a modifier.

opcode
When used as a modifier or noun.

open systems
(Or XSI-conformant systems) Use generically. If you need to specify the subset of open systems that are based on the UNIX operating system, write: ‘‘UNIX operating systems and their derivatives.’’

option
Argument to a command that (typically) changes the default behavior of the command.

option-argument
Argument to an option.

output
When used as a noun only, not as a verb.

Pascal
Note initial capital.

PID
Process identifier.

POSIX
Portable Operating System Interface for Computer Environments. Note capitalization.

PostScript
Note capitalization.

path name
When used as a noun.

pathname
When used as a variable in syntax examples.

path-name
When used as a modifier.

per

postprocessor

preprocessor

press
Used to indicate the action of pressing a key that does not echo to the screen; the Control key is one such example.

previous
Do not use to refer to another location by position.

print out
When used as a verb.

printout
When used as a noun.

pthreads
Note lowercase initial.

read-only
When used as a modifier.

realtime
When used as a modifier or noun.

reentrant
When used as a modifier.

reference page
Use instead of manpage, manual page, and so on.

remove
Use this verb to refer to a dialog box. For example, ‘‘The dialog box is removed from the screen.’’

runtime
When used as a modifier or noun.

SCCS
Source Code Control System.

SGML
Standard Generalized Markup Language.

STREAMS
Note capitalization.

s
Second.

screen button
Use to refer to a button on a screen. Use the verb *click on* for controls on the screen.

screen object
Anything that appears on a screen; for example, box, menu, icon, and so on. Do not use the names of screen objects as verbs.

sec
Second.

select
Use this verb to designate information that will be the subject of a subsequent operation.

set up
When used as a verb.

set-up
When used as a modifier.

setup
When used as a noun.

shall
Describes a feature or behavior that is mandatory for an implementation that conforms to the definitions contained in the specification. An application can rely on the existence of the feature or behavior.

shell, the
Change to ‘‘the shell as documented in the *sh*( ) reference page’’ when referring to the default shell provided by systems that conform to the Single UNIX Specification.

short-term
When used as a modifier or noun.

should
For an implementation that conforms to the definitions contained in the specification, describes a feature or behavior that is recommended but optional. An application should not rely on the existence of the feature or behavior. An application that relies on such a feature or behavior is not portable across conforming implementations.

For an application, describes a feature or behavior that is recommended programming practice for maximum portability.

shut down
When used as a verb.

shutdown
When used as a modifier.

start-up
When used as a modifier.

startup
When used as a noun.

string
Contiguous sequence of bytes, terminated by and including the first null byte.

subdirectory
When used as a noun.

superuser
When used as a noun.

sync
Synchronous.

TCP
Transmission Control Protocol.

TCP/IP
Transmission Control Protocol/Internet Protocol.

tab stop
When used as a noun.

text-only
When used as a modifier.

that
When used as a restrictive pronoun. For example, ‘‘... the subset of open systems that are based on ...’’

time out
When used as a verb.

time zone
When used as a noun.

timeout
When used as a noun.

type
Used to indicate the entering of information. For example, ‘‘Type the following command.’’

UID
User identification.

UNIX
Note capitalization.

U.S.
United States.

UUCP
UNIX-to-UNIX Copy.

undefined
Describes the nature of a value or behavior not defined by the definitions contained in the specification which will result from use of an invalid program construct or invalid data input.

The value or behavior may vary among implementations that conform to the definitions contained in the specification. An application should not rely on the existence or validity of the value or behavior. An application that relies on any particular value or behavior is not portable across conforming implementations.

unspecified
Describes the nature of a value or behavior not specified by the definitions contained in the specification which will result from use of a valid program construct or valid data input.

The value or behavior may vary among implementations that conform to the definitions contained in the specification. An application should not rely on the existence or validity of the value or behavior. An application that relies on any particular value or behavior is not portable across conforming implementations.

uppercase
When used as a modifier or noun.

user ID
When used as a noun.

user name

STREAMS
Note capitalization. When used as a noun.

utility
Executable file that can be called by name from a shell (not including *built-in utilities*). See also *command*.

versus

via
Change to "through" or "by means of."

vice versa

viz
Expand to "namely."

when
Use if an event is inevitable. Do not use to mean in contrast/comparison to.

where
Do not use to mean in contrast/comparison to.

which
When used as a nonrestrictive pronoun, and preceded with a comma. For example, "... the X/Open Portability Guide, which contains ..."

while
Do not use to mean in contrast/comparison to.

will
Same meaning as *shall*; *shall* is the preferred term.

windows
Use the verbs *open* and *close* to refer to windows.

worldwide
When used as a modifier.

write-only
When used as a modifier.

X Window System
When used as a modifier or noun.

x-axis

x-coordinate

y-axis

y-coordinate

zeros
Not zeroes.

Appendix B

Extensions

This appendix defines the extensions in use at the time of publication.

The short code in parentheses should be displayed in the output.

Extension (EX)
: The feature described is an extension to ISO POSIX-1 and ISO POSIX-2. Application writers may confidently make use of an extension as it will be supported on all XSI-conformant systems.

FIPS Requirements (FIPS)
: The **Federal Information Processing Standards (FIPS)** are a series of U.S. Government Procurement Standards managed and maintained on behalf of the U.S. Department of Commerce by the National Institute of Standards and Technology (NIST). The feature described has been restricted in order to align with the FIPS requirements.

Job Control Extension (JC)
: Job control is an optional feature in the operating system described by ISO POSIX-1, but it is supported by all XSI-conformant systems.

Obsolescent (OB)
: The feature described is obsolescent. It is fully portable to all current XSI-conformant systems, but may be withdrawn in future issues.

Output format incompletely specified (OF)
: The format of the output produced by the feature is not fully specified. It is therefore not possible to post-process this output in a consistent fashion. Typical problems include unknown length of strings and unspecified field delimiters.

Optional header (OH)
: This indicates that the marked header is not required on XSI-conformant systems.

Dependent on optional service in XSI (OP)
: Typical implementations depend on an optional service and the functionality affected need not be present if the optional service is not supported.

The behavior cannot be guaranteed to be consistent (PI)
: It is not possible to guarantee that the feature behaves in the same way on all XSI-conformant systems.

Realtime (RT)
: The feature described is part of the Realtime Feature Group.

Realtime Threads (RTT)
: The feature described is part of the Realtime Threads Feature Group.

Possibly unsupportable feature (UN)
: It need not be possible to implement the required functionality (as defined) on all XSI-conformant systems and the functionality need not be present.

Index