

IEEE Standard for Information Technology — Portable Operating System Interface (POSIX™) System Administration — Part 3: User and Group Account Administration

Sponsor
Portable Applications Standards Committee
of the
IEEE Computer Society

Approved 10 December 1996

IEEE Standards Board

Approved 5 June 1997

American National Standards Institute

Abstract: IEEE Std 1387.3-1996 System Administration Interface/User and Group Administration for Computer Operating System Environments, is part of the POSIX™ series of standards for applications and user interfaces to open systems. The purpose of this standard is to provide a common set of utility programs, for the administration of the User and Group Account entities described in the ISO/IEC 9945-1:1996 (IEEE Std 1003.1-1996) and ISO/IEC 9945-2:1993 (IEEE Std 1003.2-1992) standards.

Keywords: open systems, operating systems, portable application, POSIX, user, group, system administration.

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

*The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street, New York, NY 10017-2394, USA*

*Copyright ©1997 by the
Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 1997
Printed in the United States of America*

ISBN 1-55937-866-2

*No part of this publication may be reproduced in any form,
in an electronic retrieval system or otherwise,
without the prior written permission of the publisher.*

IEEE Standards documents are developed within the Technical Committees of the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Board. Members of the committees serve voluntarily and without compensation. They are not necessarily members of the Institute. The standards developed within IEEE represent a consensus of the broad expertise on the subject within the Institute as well as those activities outside of IEEE that have expressed an interest in participating in the development of the standard.

Use of an IEEE Standard is wholly voluntary. The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of the IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of all concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, the IEEE and the members of its technical committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE Standards Board
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331

Note: Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying patents for which a license may be required by an IEEE standard or for conducting inquiries into the legal validity or scope of those patents which are brought to its attention.

Authorization to photocopy portions of any individual standard for internal or personal use is granted by the Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to the Copyright Clearance Center. To arrange for payment of licensing fee, please contact Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923 USA; (508) 750-8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Contents

	PAGE
Introduction	iv
Section 1: General	1
1.1 Scope	1
1.2 Normative References	2
1.3 Conformance	2
1.4 Test Methods	6
Section 2: Terminology and General Requirements	7
2.1 Conventions	7
2.2 Definitions	10
2.3 Utility Conventions	15
2.4 Utility Description Defaults	15
2.5 Configuration Values	21
Section 3: User And Group Account Management Utilities	23
3.1 Common Definitions for Utilities	23
3.2 User and Group Account Option-Arguments	28
3.3 User and Group Account Extended Options	30
3.4 useradd - Add a new user account	37
3.5 userdel - Delete a user account	42
3.6 usermod - Modify user account options	45
3.7 userls - List user account options	51
3.8 groupadd - Add a new group account	55
3.9 groupdel - Delete a group account	58
3.10 groupmod - Modify a group account	61
3.11 groups - Lists options of a group	65
3.12 passwd - Change user or group password	69
Annex A (informative) Bibliography	73
Annex B (informative) Rationale and Notes	75
B.1 General	75
B.2 Terminology and General Requirements	78
B.3 Rationale - User and Group Account Management Utilities	78
Annex C (informative) Distributed System Management Issues	89
Annex D (informative) Portability Considerations	91
D.1 User Requirements	91
D.2 Portability Capabilities	92
D.3 Profiling Considerations	95
Alphabetic Topical Index	97

TABLES

Table 1-1	–	Conformance Options	4
Table 2-1	–	Typographical Conventions	7
Table B-1	–	Base Documents	77

Introduction

(This Introduction is not a normative part of IEEE Std 1387.3-1996 IEEE Standard for Information Technology — Portable Operating System Interface (POSIX™) System Administration — Part 3: User and Group Account Administration, but is included for information only.)

The purpose of this standard is to define a standard interface for the administration of users and groups. It is intended for systems implementors, system administrators, and application developers and is complementary to ISO/IEC 9945-1:1996 (IEEE Std 1003.1-1996) and ISO/IEC 9945-2:1993 (IEEE Std 1003.2-1992). This standard is based upon documentation and the knowledge of existing programs that assume an interface and architecture similar to that described by POSIX.1 {1} and POSIX.2 {2}. (See Section 1.1 for a full description of the relation-ship between these standards.)

Organization of the Standard

The standard is divided into the following parts:

- (1) General
- (2) Terminology and General Requirements
- (3) User and Group Administration Utilities

This introduction and any appendices are not considered a normative part of the standard.

Conformance Measurement

In publishing this standard, both IEEE and the IEEE Std 1387.3-1996 developers simply intend to provide a yardstick against which various operating system implementations may be measured for conformance. It is not the intent of either the IEEE or IEEE Std 1387.3-1996 developers to measure or rate any products or to reward or sanction any vendor's products as standard by these or any other means. The responsibility for determining the degree of conformance or lack thereof with this standard rests solely with the individual evaluating the product claiming to be in conformance with the standard.

Base Documents

Several sources for a base document were investigated. This included documents provided by Fermi National Accelerator Laboratory, IBM, The Open Software Foundation, The Santa Cruz Operation, SunSoft, Tivoli Systems, and Unix System Laboratories. This included presentations on existing practice by The Open Software Foundation, The Santa Cruz Operation, SunSoft, Tivoli Systems, and UNIX System Laboratories.

The Santa Cruz Operation's managed object definitions and utilities, along with the USL `user*` and `group*` commands, were deemed to comprise a functionally adequate base upon which all parties could build. While the group recognized that these did not address every concern, it was felt that that the utility descriptions (including detailed behavior), provided an excellent starting point.

39 **Extensions and Supplements to this Standard**

40 Activities to extend this standard to address additional requirements are in pro-
41 gress and similar efforts can be anticipated in the future. This is an outline of
42 how these extensions will be incorporated and how users of this document can
43 keep track of that status.

44 Extensions are approved as supplements to this document following the Institute
45 of Electrical and Electronics Engineers (IEEE) standards procedures and eventu-
46 ally the International Organization for Standardization/International Electro-
47 technical Committee (ISO/IEC) standards.

48 Approved supplements are published separately and distributed with orders from
49 the IEEE for this document until the full document is reprinted and such supple-
50 ments are incorporated in their proper positions.

51 If there are any questions about the completeness of your version, you may con-
52 tact the IEEE Computer Society, (202)371-0101, or the IEEE Standards Office,
53 (212)705-7960, to determine what supplements have been published.

54 Supplements may contain either required functions or optional facilities. Supple-
55 ments may add additional conformance requirements (see Section 2, *Confor-*
56 *mance* which defines new classes of conforming systems or applications).

57 It is undesirable (but perhaps unavoidable) for supplements to change the func-
58 tionality of the already defined facilities.

59 If interested in participating in addressing IEEE Std 1387.3-1996 issues and
60 developing the IEEE Std 1387.3-1996 standards, please send your name, address,
61 and phone number to:

62 Secretary, IEEE Standards Board
63 P.O. Box 1331
64 445 Hoes Lane
65 Piscataway, NJ 08844-1331
66 USA

67 When writing, ask to have your letter forwarded to the chairperson or appropriate
68 reviewer/developer of IEEE Std 1387.3-1996.

69 IEEE Std 1387.3-1996 was prepared by the P1387.3 working group, sponsored by
 70 the Portable Applications Standards Committee of the IEEE Computer Society.
 71 When the IEEE Standards Board approved this standard on 10 December 1996,
 72 the membership of the IEEE P1387.3 working group was as follows:

73 **Portable Application Standards Committee (PASC)**

74	Chair:	Lowell Johnson
75	Vice-Chair:	Jay Ashford
76	Vice-Chair:	Andy Bihain
77	Vice-Chair:	Andrew Josey
78	Vice-Chair:	Barry Needham
79	Treasurer:	Peter Smith
80	Secretary:	Charles Severance

81 **P1387 Working Group Officials**

82	Chair:	Martin Kirk
83		Steve Carter (1988-1990)
84	Vice-Chair:	Jay Ashford
85		David Hinnant(1988-1991)
86	Tech Editor:	
87		Matt Wicks (1991-1993)
88		Robert Robillard (1991-1993)
89		Shoshana O'Brien (1989-1991)
90		Bob Baumann (1988-1990)
91	Secretary:	Bernard Kinsler
92		John Pokladnik (1990)
93		Mark Colburn (1989)

94 **IEEE Std 1387.3-1996 Working Group Officials**

95	Chair:	Louis Imershein
96		Jeff Parker (1990-1992)
97	Tech Editor:	Michael Browne
98		Louis Imershein (1993)
99		Jeff Parker (1990-1992)
100	Secretary:	Bernard Kinsler

101 **IEEE Std 1387.3-1996 Working Group Members**

102	Michael Browne	James Fleming	Janet Frazier
103	Jay Goldberg	Brian Hackworth	Steve Hanson
104	Joseph Huntgate	Louis Imershein	Bernard Kinsler
105	Barry Needham	Jeff Parker	Peter Valdes
106	Matthew Wicks	Susanne Wilhelm	

107 **IEEE Std 1387.3-1996 Technical Reviewers**

108	Jan R. Badovinat	Michael E. Browne	Will Frazier
109	David Farley	Louis Imershein	Steve Hansen
110	Joe Hungate	Bernard Kinsler	

The following persons were members of the balloting group:

111	Norman Aaronson	Theodore P. Baker	Andy Bihain
112	Kieth Bostic	Michael E. Browne	Charles Debaun
113	Philip H. Enslow	Barry Hedquist	Joseph Hungate
114	Louis Imershein	Janecek Petr	Hal Jespersen
115	Lowell Johnson	Judy Kerner	Bernard Kinsler
116	Martin Kirk	Joshua Knight	Kenneth C. Kung
117	Thomas M. Kurihara	Richard Lagerstorm	Frederic Marand
118	Marshall McKusick	Wendy Rauch	Leonard W. Seagren
119	Thomas Shem	Mark-Rene Uchida	Matthew J. Wicks
120	Oren Yuen		

121 When the IEEE Standards Board approved this standard on 10 December 1996, it
122 had the following membership:

123 **Donald C. Loughry, *Chair*** **Richard J. Holleman, *Vice Chair***
124 **Andrew G. Salem, *Secretary***

125	Giles A. Baril	E.G. "Al" Kiener	Jose R. Ramos
126	Clyde R. Camp	Joseph L. Koepfinger*	Arther K. Reilly
127	Joseph A. Cannatelli	Stephen R. Lambert	Ronald H. Reimer
128	Stephen L. Diamond	Lawrence V. cCall	Gary S. Robinson
129	Harold E. Epstein	L. Bruce McClung	Ingo Rüsçh
130	Donald C. Fleckenstein	Marco W. Migliaro	John S. Ryan
131	Jay Forster*	Mary Lou Padgett	Chee Kiow Tan
132	Donald N. Heirman	John W. Pope	Leonard L. Tripp
133	Ben C. Johnson		Howard L. Wolfman

134 *Member Emeritus

135 Also included are the following nonvoting IEEE Standards Board liaisons:

136	Satish K. Aggarwal
137	Alan H. Cookson
138	Chester C. Taylor

139 Lisa S. Young
140 *IEEE Standards Project Editor*

IEEE Standard for Information Technology — Portable Operating System Interface (POSIX™) System Administration — Part 3: User and Group Account Administration

Section 1: General

1.1 Scope

User administration includes, but is not limited to, tasks such as the creation and maintenance of user accounts and groups in both single systems and heterogeneous distributed environments. This standard provides the distributed management for a POSIX.1 {1}¹⁾ and POSIX.2 {2} conformant system.

Although POSIX.1 {1} describes a user database, a group database, and the concept of a user's home directory, utilities to manage these entities are not described by any current POSIX standard. The POSIX User Administration working group takes as its mission the management of these entities as described in POSIX.1 {1} and POSIX.2 {2} as well as the management of optional extensions such as an account password.

1) The numbers in curly brackets correspond to those of the references in 1.2. When preceded by a "B," the numbers correspond to those of the bibliography in Annex A.

1.2 Normative References

The following standards contain provisions which, through references in this text, constitute provisions of this standard. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards.

- {1} ISO/IEC 9945-1: 1996 (IEEE Std 1003.1-1996), *Information technology—Portable Operating System Interface (POSIX™)—Part 1: System. Application Program Interface (API) [C Language]*.²⁾
- {2} ISO/IEC 9945-2: 1993 (IEEE Std 1003.2-1992), *Information technology—Portable Operating System Interface (POSIX™)—Part 2: Shell and Utilities*.³⁾

1.3 Conformance

1.3.1 Implementation Conformance

1.3.1.1 Requirements

A *conforming implementation* shall meet all of the following criteria:

- (1) The system shall support all required interfaces defined within this standard. These interfaces shall support the functional behavior described herein.
- (2) The system may provide additional or enhanced utilities, functions, or facilities not required by this standard. Nonstandard extensions should be identified as such in the system documentation. Nonstandard extensions, when used, may change the behavior of utilities, functions, or facilities defined by this standard. In such cases, the implementation's conformance document (see 2.2.1.3) shall define an execution environment (i.e., shall provide general operating instructions) in which an application can be run with the behavior specified by this standard. In no case shall such an environment require modification of a Strictly Conforming POSIX.7.3 Application.

-
- 2) ISO/IEC documents are available from the ISO Central Secretariat, Case Postale 56, 1 rue de Varembe, CH-1211, Genève 20, Switzerland/Suisse. ISO publications are also available in the United States from the Sales Department, American National Standards Institute, 11 W.42nd St., 13th Floor, New York, NY 10036, USA.
 - 3) IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

53 1.3.1.2 Documentation

54 A conformance document with the following information shall be available for an
55 implementation claiming conformance to this standard. The conformance docu-
56 ment shall have the same structure as this standard, with the information
57 presented in the appropriately numbered sections; sections that consist solely of
58 subordinate section titles, with no other information, are not required.

59 The conformance document shall not contain information about extended facilities
60 or capabilities outside the scope of this standard, unless those extensions affect
61 the behavior of a Strictly Conforming POSIX.7.3 Application; in such cases, the
62 documentation required by the previous subclause shall be included.

63 The conformance document shall contain a statement that indicates the full name,
64 number, and date of this standard that applies. The conformance document may
65 also list user and group management standards approved by ISO/IEC or any
66 ISO/IEC member body that are available for use by a Conforming POSIX.7.3 Appli-
67 cation. It should indicate whether it is based on a fully conformant POSIX.1 {1}⁴⁾
68 or POSIX.2 {2} system. Applicable characteristics where documentation is
69 required by one of these standards, or by standards of government bodies, may
70 also be included.

71 The conformance document shall describe the behavior of the implementation for
72 all implementation-defined features defined in this standard. This requirement
73 shall be met by listing these features and providing either a specific reference to
74 the system documentation or providing full syntax and semantics of these
75 features. When the value or behavior in the implementation is designed to be
76 variable or customizable on each instantiation of the system, the implementation
77 provider shall document the nature and permissible ranges of this variation.
78 When information required by this standard is related to the underlying operating
79 system and is already available in the POSIX.1 {1} or POSIX.2 {2} conformance
80 documents, the implementation need not duplicate this information in the
81 POSIX.7.3 conformance document, but may provide a cross-reference for this pur-
82 pose.

83 The conformance document may specify the behavior of the implementation for
84 those features where this standard states that implementations may vary or
85 where features are identified as undefined or unspecified. The conformance docu-
86 ment may contain information regarding the support, or non-support of the
87 optional features and behaviors identified in this standard.

88 No specifications other than those described in this subclause (1.3.1.2) shall be
89 present in the conformance document.

90 The phrase “shall be documented” in this standard means that documentation of
91 the feature shall appear in the conformance document, as described previously,
92 unless the system documentation is explicitly mentioned.

93
94 4) The numbers in curly brackets correspond to those of the references in 1.2. When preceded by a
95 “B,” the numbers correspond to those of the bibliography in Annex A.

96 The system documentation should also contain the information found in the con-
97 formance document.

98 1.3.1.3 Conforming Implementation Options

99 The conformance options specified in POSIX.7.3 are outlined in Table 1-1.

100 **Table 1-1 – Conformance Options**

101	102	103
	Name	Description
103	[P1387_3_DEFAULT]	Modifiable default attributes are supported.
104	[P1387_3_CELL]	The system is a member of a cell and a basic set of options is supported to administer user and group accounts within that cell.
105		
106		
107	[P1387_3_CELL_EXTENDED]	The system is a member of a cell and an extended set of options is supported to administer that cell.
108		[P1387_3_CELL] shall also be supported when this option is supported.
109		
110	[P1387_3_PW]	Passwords may be applied to a user account for purposes of authentication.
111		
112	[P1387_3_PW_EXTENDED]	Password aging, and password controls are supported.
113		[P1387_3_PW] shall also be supported when this option is supported.
114	[P1387_3_PW_GROUP]	Passwords may be associated with group accounts (see <code>newgrp</code> in POSIX.2 {2})
115		
116		
117	[P1387_3_ACCESS]	Options which may be used to monitor and limit user logins.
118		

119 The following utilities shall be supported by conforming implementations of
120 POSIX.7.3: `useradd`, `userdel`, `usermod`, `userls`, `groupadd`, `groupdel`,
121 `groupmod`, and `groupls`.

122 Unless explicitly designated as conditional features, all parameters for these utili-
123 ties are required.

124 The `passwd` utility shall be supported by conforming implementations of
125 POSIX.7.3, when the [P1387_3_PW] conformance option is supported.

126 Additional language bindings and development utility options may be provided in
127 other related standards or in future revisions to this standard. In the former
128 case, additional symbolic constants of the same general form as shown in this sub-
129 clause should be defined by the related standard document and made available to
130 the application, without requiring this standard to be updated.

131 1.3.2 Application Conformance

132 All applications claiming conformance to this standard fall within one of the fol-
133 lowing categories:

134 1.3.2.1 Strictly Conforming POSIX.7.3 Application

135 A Strictly Conforming POSIX.7.3 Application is an application that requires only
the facilities described in this standard. Such an application:

- 136 (1) shall accept any implementation behavior that results from actions it
137 takes in areas described in this standard as *implementation-defined* or
138 *unspecified*, or where this standard indicates that implementations may
139 vary;
- 140 (2) shall not perform any actions that are described as producing *undefined*
141 results;
- 142 (3) shall not use facilities designated as *obsolescent* ;
- 143 (4) shall tolerate, and is permitted to adapt to, the presence or absence of
144 optional facilities whose availability is indicated by using the verb *may*.

145 Within this standard, any restrictions placed upon a Conforming POSIX.7.3 Appli-
146 cation also shall restrict a Strictly Conforming POSIX.7.3 Application.

147 **1.3.2.2 Conforming POSIX.7.3 Application**

148 The term Conforming POSIX.7.3 Application is used to describe either of the two
149 following application types.

150 **1.3.2.2.1 ISO/IEC Conforming POSIX.7.3 Application**

151 An ISO/IEC Conforming POSIX.7.3 Application is an application that uses only the
152 facilities described in this standard and approved conforming language bindings
153 for any ISO/IEC standard. Such an application shall include a statement of con-
154 formance that documents all options and limit dependencies, and all other
155 ISO/IEC standards used.

156 **1.3.2.2.2 <National Body> Conforming POSIX.7.3 Application**

157 A <National Body> Conforming IEEE Std 1387.3-1996 Application differs from an
158 ISO/IEC Conforming POSIX.7.3 Application in that it also may use specific stan-
159 dards of a single ISO/IEC member body referred to here as “<National Body>.”
160 Such an application shall include a statement of conformance that documents all
161 options and limits dependencies, and all other <National Body> standards used.

162 **1.3.2.3 Conforming POSIX.7.3 Application Using Extensions**

163 A Conforming POSIX.7.3 Application Using Extensions is an application that
164 differs from a Conforming POSIX.7.3 Application only in that it uses nonstandard
165 facilities that are consistent with this standard. Such an application shall fully
166 document its requirements for these extended facilities, in addition to the docu-
167 mentation required of a Conforming POSIX.7.3 Application. A Conforming
168 POSIX.7.3 Application Using Extensions shall be either an ISO/IEC Conforming
169 POSIX.7.3 Application Using Extensions or a <National Body> Conforming
170 POSIX.7.3 Application Using Extensions (see 1.3.2.2.1 and 1.3.2.2.2).

171 **1.4 Test Methods**

172 None.

Section 2: Terminology and General Requirements

2.1 Conventions

2.1.1 Editorial Conventions

This standard uses the following editorial and typographical conventions. A summary of typographical conventions is shown in Table 2-1.

Table 2-1 – Typographical Conventions

Reference	Example
Attribute of a Structure (object)	<i>filesets</i> or <i>product.filesets</i>
C-Language Data Type	<i>long</i>
C-Language Function	<i>system()</i>
Conformance Option	[P1387_3_PW]
Cross Reference: Annex	Annex A
Cross Reference: Clause	2.3
Cross Reference: Other Standard	ISO/IEC 9999-1 { <i>n</i> }
Cross Reference: Section	Section 2
Cross Reference: Subclause	2.3.4, 2.3.4.5, 2.3.4.5.6
Defined Term	(see text)
Environment Variable	PATH
Error Number	[EINTR]
Example Input	echo Hello, World
Example Output	Hello, World
Figure Reference	Figure 7-1
File Name	/tmp
Keyword	file_permissions
Parameter	<directory pathname>
Script Types	useradd.pre
Special Character	<newline>
Symbolic Constant, Limit	{_POSIX_VDISABLE}, {LINE_MAX}
Table Reference	Table 6-1
Utility Name	awk
Utility Operand	<i>file_name</i>
Utility Option	-c
Utility Option with Option-Argument	-w <i>width</i>

The **Bold** font is used to show brackets that denote optional arguments in a utility synopsis, as in

```
cut [-c list] [file_name]
```

These brackets shall not be used by the application unless they are specifically mentioned as literal input characters by the utility description.

38 There are symbols enclosed in angle brackets (< >):

39 Parameters Parameters, also called *metavariables*, are in italics,
40 such as <directory pathname>. The entire symbol,
41 including the brackets, is meant to be replaced by the
42 value of the symbol described within the brackets.

43 Numbers within braces, such as “POSIX.1 {1},” represent cross references to the
44 Normative References clause (see 1.2). If the number is preceded by a B, it
45 represents a Bibliographic entry (see Annex A). Bibliographic entries are for
46 information only.

47 In some examples, the **Bold Courier** font is used to indicate the output of the
48 system that resulted from some user input, shown in *Courier*.

49 Defined terms are shown in three styles, depending on context:

50 (1) Terms defined in 2.2.1, and 2.2.2 are expressed as subclause titles. Alter-
51 native forms of the terms appear in [brackets]. At the conclusion of this
52 type of definition, the designation of a standard within [brackets] indi-
53 cates that the text of the definition was copied from that standard, with
54 only editorial changes, if any.

55 (2) The initial appearances of other terms, applying to a limited portion of
56 the text, are in *italics*.

57 (3) Subsequent appearances of the term are in the Roman font.

58 Symbolic constants are shown in two styles: those within curly braces are
59 intended to call the attention of the reader to values in <limits.h> and
60 <unistd.h>; those without braces are usually defined by one or a few related
61 functions. There is no semantic difference between these two forms of
62 presentation.

63 Filenames and pathnames are shown in *Courier*. When a pathname is shown
64 starting with “\$HOME/”, this indicates the remaining components of the path-
65 name are to be related to the directory named by the user’s **HOME** environment
66 variable.

67 The style selected for some of the special characters, such as <newline>,
68 matches the form of the input given to the `localedef` utility (see POSIX.2 {2},
69 section {2.5.2}). Generally, the characters selected for this special treatment are
70 those that are not visually distinct, such as the control characters <tab> or
71 <newline>.

72 Literal characters and strings used as input or output are shown in various ways,
73 depending on context:

74 %, begin When no confusion would result, the character or string is ren-
75 dered in the *Courier* font and used directly in the text.

76 ‘c’ In some cases a character is enclosed in single-quote characters,
77 similar to a character constant used by several programming
78 languages. Unless otherwise noted, the quotes shall not be used
79 as input or output.

 "string" In some cases, a string is enclosed in double-quote characters,
 similar to a character constant used by several programming

languages. Unless otherwise noted, the quotes shall not be used as input or output.

Defined names that are usually in lowercase, particularly function names, are never used at the beginning of a sentence or anywhere else that English usage would require them to be capitalized.

Parenthetical expressions within normative text also contain normative information. The general typographic hierarchy of parenthetical expressions is:

{ [()] }

The square brackets are most frequently used to enclose a parenthetical expression that contains a function name [such as *waitpid()*] with its built-in parentheses.

In some cases, tabular information is presented inline; in others it is presented in a separately labeled Table. This arrangement was employed purely for ease of reference and there is no normative difference between these two cases.

Annexes marked as *normative* are parts of the standard that pose requirements, exactly the same as the numbered Sections, but have been moved to near the end of the document for clarity of exposition. *Informative* Annexes are for information only and pose no requirements. All material preceding page 1 of the document (the “front matter”) and the two indexes at the end are also only informative.

NOTES that appear in a smaller point size have one of two different meanings, depending on their location:

— When they are within the normal text of the document, they are the same as footnotes—informative, posing no requirements on implementations, distributions, or applications.

— When they are attached to Tables or Figures, they are normative, and can include requirements.

Text marked as examples (including the use of “e.g.”) is for information only.

The table in this section refers to constructs from the C language. This standard itself has no requirement for the use of the C language or any other programming language in particular. However, this standard has been developed with the use of concepts from POSIX.1 {1} and POSIX.2 {2}, and those standards do make use of C-language constructs. The use of C-language constructs within this standard is simply to provide reference to the appropriate parts of POSIX.1 {1} and POSIX.2 {2}. As far as this standard is concerned, use of equivalent function from other languages is acceptable.

In some cases certain characters are interpreted as special characters by the shell. In the normative portions of this standard, these characters are shown without escape characters or quoting (see POSIX.2 {2}, section {3.2}). In all examples, however, quoting has been used, showing how sample commands (utility names combined with arguments) could be passed correctly to a shell (see *sh* in POSIX.2 {2}, section {4.56}). or as a string to the *system()* function.

The typographical conventions listed here are for ease of reading only. Editorial inconsistencies in the use of typography are unintentional and have no normative meaning in this standard.

2.1 Conventions

123 2.1.2 Grammar Conventions

124 Portions of this standard are expressed in terms of a special grammar notation. It
125 is used to portray the complex syntax of certain program input. The grammar is
126 based on the syntax used by the yacc utility (see POSIX.2 {2}, section {A.3}). How-
127 ever, it does not represent fully functional yacc input, suitable for program use:
128 the lexical processing and all semantic requirements are described only in textual
129 form. The grammar is not based on source used in any traditional implementa-
130 tion and has not been tested with the semantic code that would normally be
131 required to accompany it. Furthermore, there is no implication that the partial
132 yacc code presented represents the most efficient, or only, means of supporting
133 the complex syntax within the utility. Implementations may use other program-
134 ming languages or algorithms, as long as the syntax supported is the same as that
135 represented by the grammar.

136 The following typographical conventions are used in the grammar; they have no
137 significance except to aid in reading.

- 138 — The identifiers for the reserved words of the language are shown with a
139 leading capital letter. (These are terminals in the grammar. Examples:
140 While, Case.)
- 141 — The identifiers for terminals in the grammar are all named with uppercase
142 letters and underscores. Examples: NEWLINE, ASSIGN_OP, NAME.
- 143 — The identifiers for nonterminals are all lowercase.

144 2.2 Definitions

145 2.2.1 Terminology

146 For the purposes of this standard, the following definitions apply:

147 **2.2.1.1 application:** The use of capabilities provided by an information system
148 specific to the satisfaction of a set of user requirements. These capabilities include
149 hardware, software and data.

150 **2.2.1.2 can:** An indication of a permissible optional feature or behavior available
151 to the application; the implementation shall support such features or behaviors as
152 mandatory requirements.

153 **2.2.1.3 conformance document:** A document provided by an implementor that
154 contains implementation details as described in 1.3.1.2.

155 **2.2.1.4 implementation:** A set of resources, including hardware and software,
156 that support the services defined by this standard.

157 **2.2.1.5 implementation defined:** An indication that the implementation pro-
vider shall define and document the requirements for correct program constructs
and correct data of a value or behavior in a conformance document.

158 **2.2.1.6 may:** An indication of an optional feature or behavior of the implementa-
159 tion that is not required by this standard, although there is no prohibition against
160 providing it.

161 A Strictly Conforming POSIX.7.3 Application is permitted to use such features, but
162 shall not rely on the implementation's actions in such cases. To avoid ambiguity,
163 the reverse sense of *may* is not expressed as *may not*, but as *need not*.

164 **2.2.1.7 obsolescent:** An indication that a certain feature may be considered for
165 withdrawal in future revisions of this standard.

166 Obsolescent features are retained in this version because of their widespread use.
167 Their use in new applications is discouraged.

168 **2.2.1.8 shall:** An indication of a requirement on the implementation or on
169 Strictly Conforming POSIX.7.3 Applications.

170 **2.2.1.9 should:**

171 (1) With respect to implementations, an indication of an implementation
172 recommendation, but not a requirement.

173 (2) With respect to applications, an indication of a recommended program-
174 ming practice for applications and a requirement for Strictly Conforming
175 POSIX.7.3 Applications.

176 **2.2.1.10 system documentation:** All documentation provided with an imple-
177 mentation, except the conformance document.

178 **2.2.1.11 undefined:** An indication that this standard imposes no portability
179 requirements on applications for erroneous program construction, erroneous data,
180 or use of an indeterminate value.

181 Implementations may specify the result of using that value or causing that
182 behavior. An application using such behaviors is using extensions, as defined in
183 1.3.2.3.

184 **2.2.1.12 unspecified:** An indication that this standard imposes no portability
185 requirements on applications for a correct program construction or correct data.

186 Implementations may specify the result of using that value or causing that
187 behavior. An application requiring a specific behavior, rather than tolerating any
188 behavior when using that functionality, is using extensions, as defined in 1.3.2.3.

189 **2.2.2 General Terms**

190 For the purposes of this standard, the following definitions apply.

191 **2.2.2.1 account:** The entity that defines an instance of a user or group.

192 **2.2.2.2 appropriate authorization:** Implementation defined controls which
193 allow users to perform restricted tasks.

194 **2.2.2.3 appropriate privileges:** An implementation defined means of associat-
195 ing privileges with an implementation defined process with regard to the function
196 calls and function call options invoked by the utilities and customization scripts
197 defined in this standard that need special privileges.

198 **2.2.2.4 argument:** A parameter passed to a utility as the equivalent of a single
199 string in the *argv* array created by one of the POSIX.1 {1} *exec* functions.

200 See 2.10.1 and 3.9.1.1 in POSIX.2 {2}. An argument is one of the options, options-
201 arguments, or operands following the command name.

202 **2.2.2.5 authorized administrator:** A user with appropriate privileges and
203 appropriate authorization.

204 **2.2.2.6 authorized user:** A user with appropriate authorization.

205 **2.2.2.7 cell:** A domain of affiliated systems which support a common mechanism
206 for the administration of users and groups.

207 **2.2.2.8 customization:** The set of subtasks that system administrators would
208 perform to adapt an implementation to their particular system in their environ-
209 ment.

210 **2.2.2.9 default:** An implementation defined or system administrator defined
211 value for a parameter to a utility.

212 **2.2.2.10 extended option:** The command options that can be specified with the
213 -x on the command line or via a -X from within a file. Extended options may be
214 defined in defaults or files.

215 **2.2.2.11 group:** This is used to identify a group of system users.

216 **2.2.2.12 group account:** An entry in the group database.

217 **2.2.2.13 group database:** The database that contains at least the following
218 information for each group: (1) Group name, (2) Numerical group ID, and (3) List
219 of all users allowed in the group. The use of the word database does not have any
220 implications on the implementation of how this data is stored.

221 **2.2.2.14 group name:** The name associated with a particular group id in the
222 group database.

223 **2.2.2.15 group ID:** A nonnegative integer which can be contained in an object of
type *gid_t*, that is used to identify a group of system users. [POSIX.1 {1}]

- 224 **2.2.2.16 locale:** The definition of the subset of the environment of a user that
225 depends on language and cultural conventions. [POSIX.2 {2}.]
- 226 **2.2.2.17 locked:** Denying access to a user or group entity. This is an administra-
227 tive function unrelated to permissions or authentication.
- 228 **2.2.2.18 login:** The unspecified activity by which a user gains access to the sys-
229 tem. Each login shall be associated with exactly one login name. [POSIX.1 {1}]
- 230 **2.2.2.19 option:** An argument to a command that is generally used to specify
231 changes in the default behavior of a utility. [POSIX.2 {2}].
- 232 **2.2.2.20 password:** A string that is used to authenticate a group or user
233 account.
- 234 **2.2.2.21 session:** A collection of process groups established for job control pur-
235 poses. [POSIX.1 {1}.]
- 236 **2.2.2.22 password lifetime:** The amount of time a password is valid.
- 237 **2.2.2.23 self-chosen passwords:** A password that is selected by the user of a
238 group or user account.
- 239 **2.2.2.24 Standard Error:** An output stream usually intended to be used for
240 diagnostic messages. [POSIX.2 {2}]
- 241 **2.2.2.25 Standard Input:** An input stream usually intended to be used for pri-
242 mary data input. [POSIX.2 {2}]
- 243 **2.2.2.26 Standard Output:** An output stream usually intended to be used for
244 primary data output. [POSIX.2 {2}]
- 245 **2.2.2.27 subtask:** An component step in a particular task.
- 246 **2.2.2.28 system:** The smallest domain administered by POSIX.7.3. It may con-
247 sist of one or more POSIX.1 {1} and POSIX.2 {2} conformant operating systems
248 which closely interact to allow for the homogeneous administration of users and
249 groups.
- 250 **2.2.2.29 system administrator:** The person that is performing user administra-
251 tion on the system.
- 252 **2.2.2.30 system defaults:** The implementation-defined values for options in a
253 conforming POSIX.7.3 system. These values may be overridden by the system
254 administrator.

255 **2.2.2.31 system-generated passwords:** A password that is not selected by the
256 user of a group or user account. It is selected by a system provided utility.

257 **2.2.2.32 task:** A unit of work that an administrator performs.

258 **2.2.2.33 user:** An account on the system.

259 **2.2.2.34 user account administration:** The tasks of creation and maintenance
260 of user and group accounts. This includes creating, modifying and listing entries
261 in the user and group databases.

262 **2.2.2.35 user database:** The database that contains at least the following infor-
263 mation for each user: (1) User name, (2) Numerical user ID, (3) Numerical group
264 ID, (4) Initial working directory, and (5) Initial user program The use of the word
265 database does not have any implications on the implementation of how this data is
266 stored.

267 **2.2.2.36 user ID:** A nonnegative integer which can be contained in an object of
268 type uid_t, that is used to identify a system user. [POSIX.1 {1}.]

269 **2.2.2.37 user name:** A string that is used to identify a user. [POSIX.1 {1}.]

270 **2.2.2.38 utility:** A program that can be called by name from a shell to perform a
271 specific task or a related set of tasks. [POSIX.2 {2}.]

272 **2.2.3 Abbreviations**

273 For the purposes of this standard, the following abbreviations apply.

274 **2.2.3.1 POSIX.1:** ISO/IEC 9945-1: 1990. *Information technology—Portable*
275 *Operating System Interface (POSIX)—Part 1: System Application Program Inter-*
276 *face (API) [C Language]*

277 **2.2.3.2 POSIX.2:** ISO/IEC 9945-2: 1993. *Information technology—Portable*
278 *Operating System Interface (POSIX)—Part 2: Shell and Utilities*

279 **2.2.3.3 POSIX.7.3:** *IEEE Std 1387.3-1996, IEEE Standard for Information Tech-*
280 *nology — Portable Operating System Interface (POSIX®) System Administration*
281 *— Part 3: User and Group Account Administration*

282 **2.3 Utility Conventions**

283 The utility conventions used in this standard are the same as those described in
284 Section 2.10 of POSIX.2 {2}.

285 **2.4 Utility Description Defaults**

286 This clause describes all of the subclauses used within the utility clauses in Sec-
287 tion 3 and the other sections that describe standard utilities. It describes

- 288 (1) The intended usage of the subclause
- 289 (2) Global defaults that affect all the standard utilities.

290 **2.4.1 Synopsis**

291 The Synopsis subclause summarizes the syntax of the calling sequence for the
292 utility, including options, option-arguments, and operands. Standards for utility
293 naming are described in section 2.10.2 of POSIX.2 {2}; for describing the arguments
294 to the utility, in 2.10.1 of POSIX.2 {2}.

295 **2.4.2 Description**

296 The Description subclause describes the actions of the utility. Most explanations
297 of optional functionality are omitted here, as they are usually explained in the
298 Options subclause.

299 Some utilities in this standard are described in terms of equivalent POSIX.1 {1}
300 functionality. As explained in 1.1, a fully conforming POSIX.1(8) base is a prere-
301 quisite for this standard. The treatment of errors and intermediate results from
302 the individual functions cited are generally not specified by this standard. See Exit
303 Status and Consequences of Errors sub- clauses for the utility for all actions asso-
304 ciated with errors encountered by the utility.

305 **2.4.3 Options**

306 The Options subclause describes the utility options and option- arguments and
307 how they modify the actions of the utility. Standard utilities that have options
308 either fully comply with the Utility Syntax Guidelines in POSIX.2 {2} section 2.10.2
309 or describe all deviations. Apparent disagreements between functionality descrip-
310 tions in the Options and Description subclauses shall be resolved in favor of the
311 Options subclause.

312 Each Options subclause that uses the phrase "The ... utility shall conform to the
313 utility argument syntax guidelines ..." refers only to the use of the utility as
314 specified by this standard; implementation extensions should also conform to the
315 guidelines, but may allow exceptions for historical practice.

316 Unless otherwise stated in the utility description, when given an option that is not
 recognized by the implementation, or when a required option-argument is not pro-
 vided, standard utilities shall issue a diagnostic message to standard error and

317 exit with a nonzero exit status.

318 Default Behavior: When this subclause is listed as "None," it means that the
319 implementation need not support any options. Standard utilities that do not
320 accept options, but that do accept operands, shall recognize "--" as the first argu-
321 ment to be discarded.

322 **2.4.4 Operands**

323 The Operands subclause describes the utility operands and how they affect the
324 actions of the utility. Apparent disagreements between functionality descriptions
325 in the Operands and Description subclauses shall be resolved in favor of the
326 Operands subclause.

327 If an operand naming a file can be specified as -, which means to use the standard
328 input instead of a named file, this is explicitly stated in this subclause. Unless
329 otherwise stated, the use of multiple instances of - to mean standard input in a
330 single command produces unspecified results.

331 Unless otherwise stated, the standard utilities that accept operands shall process
332 those operands in the order specified in the command line.

333 Default Behavior: When this subclause is listed as "None," it means that the
334 implementation need not support any operands.

335 **2.4.5 External Influences**

336 The External Influences subclause describes all input data that is specified by the
337 invoker, data received from the environment, and other files or databases that
338 may be used by the utility. There are four subclauses that contain all the substan-
339 tive information about external influences; because of this, this level of header is
340 usually left blank.

341 **2.4.5.1 Standard Input**

342 The Standard Input subclause describes the standard input of the utility. This
343 subclause is frequently just a reference to the following subclause because many
344 utilities treat standard input and input files in the same manner. Unless other-
345 wise stated, all restrictions described in Input Files shall apply to this subclause
346 as well. Use of a terminal for standard input may cause any of the standard utili-
347 ties that read standard input to stop when used in the background. For this rea-
348 son, applications should not use interactive features in scripts to be placed in the
349 back-ground.

350 The specified standard input format of the standard utilities shall not depend on
351 the existence or value of the environment variables defined in this standard,
352 except as provided by this standard.

353 Default Behavior: When this subclause is listed as "None," it means that the stan-
354 dard input shall not be read when the utility is used as described by this standard.

355 2.4.5.2 Input Files

356 The Input Files subclause describes the files, other than the standard input, used
357 as input by the utility. It includes files named as operands and option-arguments
358 as well as other files that are referred to, such as startup/initialization files, data-
359 bases, etc. Commonly used files are generally described in one place and cross-
360 referenced by other utilities.

361 Some of the standard utilities, such as filters, process input files a line or a block
362 at a time and have no restrictions on the maximum input file size. Some utilities
363 may have size limitations that are not as obvious as file space or memory limita-
364 tions. Such limitations should reflect resource limitations of some sort, not arbi-
365 trary limits its set by implementors. Implementations shall define in the confor-
366 mance documentation those utilities that are limited by constraints other than file
367 system space, available memory, and other limits specifically cited by this stan-
368 dard; identify what the constraint is; and indicate a way of estimating when the
369 constraint would be reached.

370 When a standard utility reads a seekable input file and terminates without an
371 error before it reaches end-of-file, the utility shall ensure that the file offset in the
372 open file description is properly positioned just past the last byte processed by the
373 utility. For files that are not seekable, the state of the file offset in the open file
374 description for that file is unspecified.

375 When an input file is described as a text file, the utility produces undefined results
376 if given input that is not from a text file, unless otherwise stated. The utility need
377 not be able to accumulate more than {LINE_MAX} bytes from a set of multiple,
378 continued input lines. If a utility using the escaped <newline> convention
379 detects an end-of-file condition immediately after an escaped <newline>, the
380 results are unspecified.

381 Default Behavior: When this subclause is listed as "None," it means that no input
382 files are required to be supplied when the utility is used as described by this stan-
383 dard.

384 2.4.5.3 Environment Variables

385 The Environment Variables subclause lists what variables affect the execution of
386 the utility.

387 The entire manner in which environment variables described in this standard
388 affect the behavior of each utility is described in the Environment Variables sub-
389 clause for that utility, in conjunction with the global effects of the **LANG** and
390 **LC_ALL** environment variables described in POSIX.2 {2} section 2.6. The
391 existence or value of environment variables described in this standard shall not
392 otherwise affect the specified behavior of the standard utilities. Any effects of the
393 existence or value of environment variables not described by this standard upon
394 the standard utilities are unspecified.

395 Default Behavior: When this subclause is listed as "None," it means that the
396 behavior of the utility is not directly affected by environment variables described
397 by this standard when the utility is used as described by this standard.

398 **2.4.5.4 Asynchronous Events**

399 The Asynchronous Events subclause lists how the utility reacts to such events as
400 signals and what signals are caught.

401 Default Behavior: When this subclause is listed as "Default," or it refers to "the
402 standard action for all other signals"; it means that the action taken as a result of
403 the signal shall be one of the following:

- 404 (1) The action is that inherited from the parent according to the rules of
405 inheritance of signal actions defined in POSIX.1 (8) (see 2.9.1), or
- 406 (2) When no action has been taken to change the default, the default action
407 is that specified by POSIX.1 (8), or
- 408 (3) The result of the execution of the utility is as if default actions had been
409 taken.

410 **2.4.6 External Effects**

411 The External Effects subclause describes the effects of the utility on the opera-
412 tional environment, including the file system. There are three subclauses that
413 contain all the substantive information about external effects; because of this, this
414 level of header is usually left blank.

415 **2.4.6.1 Standard Output**

416 The Standard Output subclause describes the standard output of the utility. This
417 subclause is frequently just a reference to the following subclause, Output Files,
418 because many utilities treat standard output and output files in the same manner.

419 Use of a terminal for standard output may cause any of the standard utilities that
420 write standard output to stop when used in the background. For this reason,
421 applications should not use interactive features in scripts to be placed in the back-
422 ground.

423 The specified standard output of the standard utilities shall not depend on the
424 existence or value of the environment variables defined in this standard, except as
425 provided by this standard.

426 Default Behavior: when this subclause is listed as "None," it means that the stan-
427 dard output shall not be written when the utility is used as described by this stan-
428 dard.

429 **2.4.6.2 Standard Error**

430 The Standard Error subclause describes the standard error output of the utility.
431 Only those messages that are purposely sent by the utility are described.

432 Use of a terminal for standard error may cause any of the standard utilities that
433 write standard error output to stop when used in the background. For this reason,
434 applications should not use interactive features in scripts to be placed in the back-
435 ground.

436 The format of diagnostic messages for most utilities is unspecified, but the
437 language and cultural conventions of diagnostic and informative messages whose
438 format is unspecified by this standard should be affected by the setting of
439 **LC_MESSAGES**.

440 The specified standard error output of standard utilities shall not depend on the
441 existence or value of the environment variables defined in this standard, except as
442 provided by this standard.

443 Default Behavior: when this subclause is listed as "Used only for diagnostic mes-
444 sages," it means that, unless otherwise stated, the diagnostic messages shall be
445 sent to the standard error only when the exit status is nonzero and the utility is
446 used as described by this standard.

447 When this subclause is listed as "None," it means that the standard error shall not
448 be used when the utility is used as described in this standard.

449 **2.4.6.3 Output Files**

450 The Output Files subclause describes the files created or modified by the utility.
451 Temporary or system files that are created for internal usage by this utility or
452 other parts of the implementation (spool, log, audit files, etc.) are not described in
453 this, or any, subclause. The utilities creating such files and the names of such files
454 are unspecified. If applications are written to use temporary or intermediate files,
455 they should use the **TMPDIR** environment variable, if it is set and represents an
456 accessible directory, to select the location of temporary files.

457 Implementations shall ensure that temporary files, when used by the standard
458 utilities, are named so that different utilities or multiple instances of the same
459 utility can operate simultaneously without regard to their working directories, or
460 any other process characteristic other than process ID. There is one exception to
461 this requirement:

- 462 (1) Resources for temporary files other than the namespace (for example,
463 disk space, available directory entries, or number of processes allowed)
464 are not guaranteed.

465 Any temporary files created by the implementation shall be removed by the imple-
466 mentation upon successful exit of a utility, exit because of errors, or before termi-
467 nation by any of the SIGHUP, SIGINT, or SIGTERM signals, unless specified oth-
468 erwise by the utility description.

469 Default Behavior: When this subclause is listed as "None," it means that no files
470 are created or modified as a consequence of direct action on the part of the utility
471 when the utility is used as described by this standard. However, the utility may
472 create or modify system files, such as log files, that are outside of the normal exe-
473 cution environment of the utility.

474 **2.4.7 Exit Status**

475 The Exit Status subclause describes the values the utility shall return to the cal-
476 ling program (or shell) and the conditions that cause these values to be returned.
477 Usually, utilities return zero for successful completion and values greater than
478 zero for various error conditions. If specific numeric values are listed in this sub-
479 clause, conforming implementations shall use those values for the errors
480 described. In some cases, status values are listed more loosely, such as ">0." A
481 Strictly Conforming POSIX.7.3 Application shall not rely on any specific value in
482 the range shown and shall be prepared to receive any value in the range.

483 Unspecified error conditions may be represented by specific values not listed in
484 this standard.

485 **2.4.8 Consequences of Errors**

486 The Consequences of Errors subclause describes the effects on the environment,
487 file systems, process state, etc., when error conditions occur. It does not describe
488 error messages produced or exit status values used.

489 The many reasons for failure of a utility are generally not specified by the utility
490 descriptions. Utilities may terminate prematurely if they encounter invalid usage
491 of options, arguments, or environment variables; invalid usage of the complex syn-
492 taxes expressed in Extended Options; difficulties accessing, creating, reading, or
493 writing files; or difficulties associated with the privileges of the process.

494 The following shall apply to each utility, unless otherwise stated:

- 495 — If the requested action cannot be performed on an operand representing a
496 file, directory, user, process, etc., the utility shall issue a diagnostic mes-
497 sage to standard error and continue processing the next operand in
498 sequence, but the final exit status shall be returned as nonzero.
- 499 — If the requested action characterized by an option or option-argument can-
500 not be performed, the utility shall issue a diagnostic message to standard
501 error, and the exit status returned shall be nonzero.
- 502 — When an unrecoverable error condition is encountered, the utility shall exit
503 with a nonzero exit status.
- 504 — A diagnostic message shall be written to standard error whenever an error
505 condition occurs.

506 Default Behavior: when this subclause is listed as "Default," it means that
507 any changes to the environment are unspecified.

508 **2.5 Configuration Values**

509 None.

1 **Section 3: User And Group Account Management Utilities**

2 This section defines the utilities for User and Group account administration that
3 shall be implemented in all conforming POSIX.7.3 systems.

4 **3.1 Common Definitions for Utilities**

5 The utilities defined in this standard share many common definitions and
6 behaviors. This section describes the common definitions.

7 **3.1.1 Synopsis**

8 *utility_name* [*optionsString*] [-K *custom_dir*] [-x *extendedOptionString*]
9 [-X *optionsFile*] [*name*]

10 **3.1.2 Description**

11 The standard defines the following utilities: `groupadd`, `groupdel`, `groups`,
12 `groupmod`, `passwd`, `useradd`, `userdel`, `userls`, and `usermod`. Detailed
13 descriptions of these utilities are found later in this section.

14 **3.1.3 Common Extended Options Syntax**

15 The syntax for *extendedOptionString* arguments is intended for use in all imple-
16 mentations supporting this standard. The contents of *extendedOptionString* are
17 the same for options specified on the command line and for those specified in the
18 *optionsFile*.

19 The extended option string is used to set a series of options to particular values.
20 There are only a few syntax variations required for conformance with the
21 POSIX.7.3 utilities. Any particular option can be set to a system default value by
22 using `==`, as in this example

23 `userdel -x cell== username`

24 which will delete *username* from the default cell.

25 An option can be set to a specific value with the `=` syntax. As in the following
26 example:

27 `useradd -x passwd_expiration_time=3 username`
 adds the user with a password that expires in 3 days.

28 Values can be added to or subtracted from an option list with the + and - syntax as
29 in the following example:

30 `usermod -x groups+= groupname username`

31 `usermod -x groups-= groupname2 username`

32 will add the group *groupname* to the list of groups to which user
33 *username* belongs and delete the group *groupname2*. This
34 simplifies addition and deletion of values from a list of options,
35 since the previous members of the list don't have to be entered
36 again.

37 The following is the yacc syntax notation for the POSIX.7.3 extended option
38 string syntax:

```
39 %token WHITE_SPACE          /* as defined in POSIX.2 {2.2.2.191} */
40 %token SHELL_TOKEN_STRING    /* as defined in POSIX.2 {3.3}      */
41 %token PORTABLE_CHARACTER_STRING /* as defined in POSIX.2 {2.4}    */

42 %start sysadmin_option      /* IEEE Std 1387.3-1996 extended option string */
43 %%

44 sysadmin_option             : option operator_value
45                             ;

46 option                      : keyword op_ws
47                             ;

48 operator_value              : '=='
49                             | value_qualifier '=' value
50                             ;

51 value_qualifier              : /* empty */
52                             | '+' | '-'
53                             ;

54 value                       : op_ws value ws single_value
55                             | op_ws single_value
56                             ;

57 single_value                : SHELL_TOKEN_STRING
58                             ;

59 keyword                     : SHELL_TOKEN_STRING
60                             ;

61 op_ws                       : /* empty */
62                             | ws
63                             ;

64 ws                          : WHITE_SPACE
65                             ;

66 %start command_line_options
67 %%
68 command_line_options         : command_line_options ws sysadmin_option
69                             | sysadmin_option
                             ;
```

```

70  %start options_file
71  %%
72  options_file                : options_file NEWLINE option_file_line
73                             | option_file_line
74                             ;

75  option_file_line            : op_ws op_comment
76                             | op_ws sysadmin_option op_ws op_comment
77                             ;

78  op_comment                  : /* empty */
79                             | '#' PORTABLE_CHARACTER_STRING
80                             ;

```

- 81 (1) For options that support multiple values, values can be added to the
82 existing list of values by using the += operator. Similarly, values can be
83 removed by using the -= operator. Any option can be set to the default
84 value by using the == operator.
- 85 (2) A shell token can be an unquoted string or a quoted string according to
86 the rules of token recognition described in POSIX.2 {2}, section 3.3. For
87 example, it can use single or double quotes and can contain literal quotes
88 if escaped with backslash. It can also support the same level of interna-
89 tionalization as the POSIX shell.
- 90 (3) If the extended option specification contains any spaces at all, then the
91 entire specification shall be quoted if used on a command line with a
92 POSIX.2 {2} conformant shell. If the shell is not POSIX.2 {2} conformant then
93 the entire specification may be quoted.
- 94 (4) When specified on the command line, multiple -x and -X options can be
95 specified for one or more option specifications and files. Each -x and -X
96 option shall be treated as a separate command argument.
- 97 (5) When specified on the command line, multiple option specifications can
98 be included after a single -x option separated by spaces.
- 99 (6) For options files, blank lines and all comment text (any sequence of char-
100 acters beginning with an unescaped # (pound) and continuing through the
101 end of that line) are ignored according to the shell token recognition rules
102 as described in POSIX.2 {2}, section 3.3.

103 3.1.4 Default System User Management Behavior

104 For all options except for the user name or group name, if an option is not expli-
105 citly specified, then that option shall be set to an implementation defined system
106 default value. For example, the command `useradd fred` adds a user named *fred*
107 using system default values for all required options, provided that all such option
108 values can be satisfied in this case.

109 **3.1.5 Common Operand Format**

110 The group and user operands for the POSIX User and Group Account Utilities
111 shall have the following syntax:

112 [*system:*] *name*

113 Here, *name* is the name of the user or group account and *system:* is an optional
114 designation for the particular system that the utility acts on.

115 **3.1.6 Common Customization Scripts**

116 The `-K custom_dir` option is used with the utilities `useradd`, `userdel`, `user-`
117 `mod`, `groupadd`, `groupdel`, and `groupmod`. This option specifies the location of
118 scripts which are executed either prior to, or following, the specified modification
119 of system database entries by the utilities. Scripts to be used in this fashion shall
120 use the following naming convention:

121 — `utility_name.pre`

122 The utility shall perform any error checking prior to the execution of the
123 script. This script shall be executed prior to the utility modifying anything
124 on the system. If this script exits in an error, the utility shall not perform
125 any further execution.

126 — `utility_name.post`

127 This script is executed after the utility has completed modifying the system
128 and no error has occurred. The script shall not be executed when an error
129 occurs. The action taken if this script exits in an error shall be implemen-
130 tation defined.

131 The customization scripts should not be interactive and must execute without
132 user intervention.

133 **3.1.6.1 Environment Variables**

134 The following environment variables shall be set for all POSIX.7.3 utilities except
135 for `passwd` and passed to the customization scripts. The only other environment
136 variables that shall be passed to the customization scripts shall be the variables
137 defined by the individual utilities. Any other environment variables in the user's
138 environment shall not be passed to the customization scripts. The options associ-
139 ated with these environment variables are discussed in section 3.2.

140 **UG_PW_UID**

141 The user ID of the user taken from the `pw_uid` option.

142 **UG_NAME**

143 The string that uniquely identifies the user instance taken from
144 the `pw_name` option.