IEEE P1003.2 Draft 11.2 – September 1991

Copyright © 1991 by the
Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street
New York, NY 10017, USA
All rights reserved as an unpublished work.

This is an unapproved and unpublished IEEE Standards Draft, subject to change. The publication, distribution, or copying of this draft, as well as all derivative works based on this draft, is expressly prohibited except as set forth below.

Permission is hereby granted for IEEE Standards Committee participants to reproduce this document for purposes of IEEE standardization activities only, and subject to the restrictions contained herein.

Permission is hereby also granted for member bodies and technical committees of ISO and IEC to reproduce this document for purposes of developing a national position, subject to the restrictions contained herein.

Permission is hereby also granted to the preceding entities to make limited copies of this document in an electronic form only for the stated activities.

The following restrictions apply to reproducing or transmitting the document in any form: 1) all copies or portions thereof must identify the document's IEEE project number and draft number, and must be accompanied by this entire notice in a prominent location; 2) no portion of this document may be redistributed in any modified or abridged form without the prior approval of the IEEE Standards Department.

Other entities seeking permission to reproduce this document, or any portion thereof, for standardization or other activities, must contact the IEEE Standards Department for the appropriate license.

Use of information contained in this unapproved draft is at your own risk.

IEEE Standards Department
Copyright and Permissions
445 Hoes Lane, P.O. Box 1331
Piscataway, NJ 08855-1331, USA
+1 (908) 562-3800
+1 (908) 562-1571 [FAX]

255 `-l f` This library contains all library functions referenced in ISO 1539
256 {2}. An implementation shall not require this operand to be
257 present to cause a search of this library.

258 In the absence of options that inhibit invocation of the link editor, such as `-c`, the
259 `fort77` utility shall cause the equivalent of a `-l f` operand to be passed to the
260 link editor as the last `-l` operand, causing it to be searched after all other object
261 files and libraries are loaded.

262 It is unspecified whether the library `libf.a` exists as a regular file. The imple-
263 mentation may accept as `-l` operands names of objects that do not exist as regu-
264 lar files.

265 ### C.2.7.2 External Symbols

266 The FORTRAN compiler and link editor shall support the significance of external 1
267 symbols up to a length of at least 31 bytes. The compiler may fold case (i.e., may 1
268 ignore uppercase/lowercase distinctions between identifiers). The action taken 1
269 upon encountering symbols exceeding the implementation-defined maximum sym-
270 bol length is unspecified.

271 The compiler and link editor shall support a minimum of 511 external symbols
272 per source or object file, and a minimum of 4095 external symbols total. A diag-
273 nostic message is written to standard output if the implementation-defined limit
274 is exceeded; other actions are unspecified.

275 ### C.2.8 Exit Status

276 The `fort77` utility shall exit with one of the following values:

277 0 Successful compilation or link edit.

278 >0 An error occurred.

279 ### C.2.9 Consequences of Errors

280 When `fort77` encounters a compilation error, it shall write a diagnostic to stan-
281 dard error and continue to compile other source code operands. It shall return a
282 nonzero exit status, but it is implementation defined whether an object module is
283 created. If the link edit is unsuccessful, a diagnostic message shall be written to
284 standard error, and `fort77` shall exit with a nonzero status.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

C.2.10 Rationale. *(This subclause is not a part of P1003.2)*

Examples, Usage

The following are examples of usage:

`fort77 -o foo xyz.f` Compiles `xyz.f` and creates the executable `foo`.

`fort77 -c xyz.f` Compiles `xyz.f` and creates the object file `xyz.o`.

`fort77 xyz.f` Compiles `xyz.f` and creates the executable `a.out`.

`fort77 xyz.f b.o` Compiles `xyz.f`, links it with `b.o`, and creates the executable `a.out`.

History of Decisions Made

The file inclusion and symbol definition (`#define`) mechanisms used by the `c89` utility were not included in POSIX.2—even though they are commonly implemented—since there is no requirement that the FORTRAN compiler use the C preprocessor.

The `-onetrip` option was not included in this specification, even though many historical compilers support it, because it is a relic from FORTRAN-66; it is an anachronism that should not be perpetuated.

Some implementations produce compilation listings. This aspect of FORTRAN has been left unspecified because there was opposition within the balloting group to the various methods proposed for implementing it: a `-V` option overlapped with historical vendor practice and a naming convention of creating files with `.l` suffixes collided with historical `lex` file naming practice.

There is no `-I` option in this version of POSIX.2 to specify a directory for file inclusion. An INCLUDE directive has been a part of the FORTRAN-8X discussions, but it is not clear whether it will be retained.

It is noted that many FORTRAN

```
;
collation_order         : collation_order collation_entry
                        | collation_entry
collation_entry         : COLLSYMBOL EOL not chosen to avoid collision with his-
```

was

toricalimplementations.

A reference to MIL-STD-1753 was removed from an earlier draft in response to a request from the POSIX.9 working group. It was not the intention of this document to require certification of the FORTRAN compiler and the forthcoming POSIX.9 standard does not specify the military standard or any special preprocessing requirements. Furthermore, use of that document would have been inappropriate for an international standard.

The specification of optimization has been subject to changes through early drafts. At one time, `-O` and `-N` were Booleans: optimize and do not optimize (with an

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

unspecified default). Some historical practice lead this to be changed to:

`-O 0` No optimization.

`-O 1` Some level of optimization.

`-O` *n* Other, unspecified levels of optimization.

It is not always clear whether ''good code generation'' is the same thing as optimization. Simple optimizations of local actions do not usually affect the semantics of a program. The `-O 0` option has been included to accommodate the very fussy nature of scientific calculations in a highly optimized environment; compilers make errors. Some degree of optimization is expected, even if it is not documented here, and the ability to shut it off completely could be important when porting an application. An implementation may treat `-O 0` as ''do less than normal'' if it wishes, but this is only meaningful if any of the operations it performs can affect the semantics of a program. It is highly dependent on the implementation whether doing less than normal makes sense. It is not the intent of this to ask for sloppy code generation, but rather to assure that any semantically visible optimization is suppressed.

The specification of standard library access is consistent with the C compiler specification. Implementations are not required to have `/usr/lib/libf.a`, as many historical implementations do, but if not they are required to recognize `'f'` as a token.

External symbol size limits are in a normative subclause; portable applications need to know these limits. However, the minimum maximum symbol length should be taken as a constraint on a portable application, not on an implementation, and consequently the action taken for a symbol exceeding the limit is unspecified. The minimum size for the external symbol table was added for similar reasons.

The Consequences of Errors subclause clearly specifies the compiler's behavior when compilation or link-edit error occur. The behavior of several historical implementations was examined, and the choice was made to be silent on the status of the executable, or `a.out`, file in the face of compiler or linker errors. If a linker writes the executable file, then links it on disk with *lseek* ()s and *write* ()s, the partially-linked executable can be left on disk and its execute bits turned off if the link edit fails. However, if the linker links the image in memory, or if the compiler invokes the linker with options separately is

to reserve the multicharacter option namespace for vendor-specific compiler options, which are known to exist in many historical implementations. Implementations are not required to recognize, for example, `–gc` as if it were `–g –c`; nor

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

are they forbidden from doing so. The synopsis shows all of the options separately to highlight this requirement on applications.

Echoing filenames to standard error is considered a diagnostic message, because it would otherwise difficult to associate an error message with the erring file. They are describing with ‘‘may’’to allow implementations to use other methods of identifying files and to parallel the description in `c89`.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

Annex D
(inform ative)

Bibliography

{B1} ISO 639: 1988, *Code for the representation of names of languages.* [1)]

{B2} ISO 2022: 1986, *Information processing— ISO 7-bit and 8-bit coded character sets—Code extension techniques.*

{B3} ISO 2047: 1975, *Information processing—Graphical representations for the control characters of the 7-bit coded character set.*

{B4} ISO 3166: 1988, *Code for the representation of names of countries.*

{B5} ISO 6429: 1988, *Information processing—Control functions for 7-bit and 8-bit coded character sets.*

{B6} ISO 6937-2: 1983, *Information processing—Coded character sets for text communication—Part 2: Latin alphabetic and non-alphabetic graphic characters.*

{B7} ISO 8802-3: 1989, *Information processing systems—Local area networks—Part 3: Carrier sense multiple access with collision detection (CSMA / CD) access method and physical layer specification.*

{B8} ISO 8806: 1988, *Data elements and interchange formats—Information interchange —Representation of dates and times.*

{B9} ISO 8859, *Information processing—8-bit single-byte coded graphic character sets.* (Parts 1 to 8 published.)

{B10} ISO/IEC 10367: . . . ,[2)] *Information processing—Repertoire of standardized coded graphic character sets for use in 8-bit codes.*

{B11} ISO/IEC 10646: . . . ,[3)] *Information technology—Universal Coded Character Set (UCS).*

---

1) ISO documents can be obtained from the ISO office, 1, rue de Varembe´, Case Postale 56, CH-1211, Genè ve 20, Switzerland/Suisse.

2) To be approved and published.

3) To be approved and published.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

{B12} International Organization for Standardization/Association Française de Normalisation. *Dictionary of Computer Science/Dictionnaire de L'Informatique.* Geneva/Paris: ISO/AFNOR, 1989.

{B13} ANSI X3.43-1986, [4] *Representations for Local Times of the Day for Information Interchange.*

{B14} GB 2312-1980, Chinese Association for Standardization. *Coded Chinese Graphic Character Set for Information Interchange.*

{B15} JIS X0208-1990, Japanese National Committee on ISO/IEC JTC 1/SC 2. *Japanese Graphic Character Set for Information Interchange.*

{B16} JIS X0212-1990, Japanese National Committee on ISO/IEC JTC 1/SC 2. *Supplementary Japanese Graphic Character Set for Information Interchange.*

{B17} KS C 5601-1987, Korean Bureau of Standards. *Korean Graphic Character Set for Information Interchange.*

{B18} IEEE Std 100-1988, *IEEE Standard Dictionary of Electrical and Electronics Terms.*

{B19} IEEE P1003.3, [5] *Standard for Information Technology—Test Methods for Measuring Conformance to POSIX*

{B20} IEEE P1003.3.2, [6] *Standard for Information Technology—Test Methods for Measuring Conformance to POSIX. 2*

{B21} Aho, Alfred V., Kernighan, Brian W., Weinberger, Peter J., *The AWK Programming Language*, Reading, MA: Addison-Wesley, 1988.

{B22} Aho, Alfred V., Sethi, Ravi, Ullman, Jeffrey D., *Compilers, Principles, Techniques, and Tools*, Reading, MA: Addison-Wesley, 1986.

{B23} Aho, Alfred V., Ullman, Jeffrey D., *Principles of Compiler Design*, Reading, MA: Addison-Wesley, 1977.

{B24} American Telephone and Telegraph Company. *System V Interface Definition (SVID), Issues 2 and 3.* Morristown, NJ: UNIX Press, 1986, 1989. [7]

{B25} Bolsky, Morris I., Korn, David G., *The KornShell Command and Programming Language*, Englewood Cliffs, NJ: Prentice Hall, 1988.

---

4) ANSI documents can be obtained from the Sales Department, American National Standards Institute, 1430 Broadway, New York, NY 10018.

5) To be approved and published.

6) To be approved and published.

7) This is one of several documents that represent an industry specification in an area related to POSIX. 2. The creators of such documents may be able to identify newer versions that may be interesting.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

{B26} DeRemer, Frank, and Thomas J. Pennello, ''Efficient Computation of LALR(1) Look-ahead Sets.'' *SigPlan Notices* 15:8, 176-187, August, 1979.

{B27} Knuth, D. E. ''On the translation of languages from left to right.'' *Information and Control* 8:6, 607-639.

{B28} University of California at Berkeley—Computer Science Research Group. *4.3 Berkeley Software Distribution, Virtual VAX-11 Version.* Berkeley, CA: The Regents of the University of California, April 1986.

{B29} /usr/group Standards Committee. *1984 /usr/group Standard.* Santa Clara, CA: UniForum, 1984.

{B30} X/Open Company, Ltd. *X/Open Portability Guide, Issue 2.* Amsterdam: Elsevier Science Publishers, 1987.

{B31} X/Open Company, Ltd. *X/Open Portability Guide, Issue 3.* Englewood Cliffs, NJ: Prentice-Hall, 1989.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

Annex E
(inform ative)

Rationale and Notes

This annex summarizes the deliberations of the IEEE P1003.2 Working Group, the committee charged by the IEEE Computer Society's Technical Committee on Operating Systems and Operational Environments with devising an interface standard for a shell and related utilities to support and extend POSIX. 1.

The annex is being published along with the standard to assist in the process of review. It contains historical information concerning the contents of the standard and why features were included or discarded by the Working Group. It also contains notes of interest to application programmers on recommended programming practices, emphasizing the consequences of some aspects of the standard that may not be immediately apparent.

Just as this standard relies on the knowledge of architecture, history, and definitions from the POSIX. 1, so does this annex. The reader is referred to the Rationale and Notes appendix of POSIX. 1 for background material and bibliographic information about UNIX systems in general and POSIX specifically, which will not be duplicated here.

E.1 General

Editor's Note: The text of the Rationale for this section has been temporarily located in Section 1, adjacent to the text it is explaining. The text will return to this annex after the completion of balloting.

E.1.1 Scope

E.1.2 Normative References

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

E.1.3 Conformance

E.2 Terminology and General Requirements

Editor's Note: The text of the Rationale for this section has been temporarily located in Section 2, adjacent to the text it is explaining. The text will return to this annex after the completion of balloting.

E.2.1 Conventions

E.2.2 Definitions

E.2.3 Built-in Utilities

E.2.4 Character Set

E.2.5 Locale

E.2.6 Environment Variables

E.2.7 Required Files

E.2.8 Regular Expression Notation

E.2.9 Dependencies on Other Standards

E.2.10 Utility Conventions

E.2.11 Utility Description Defaults

E.2.12 File Format Notation

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

39 **E.2.13 Configuration Values**

40 ## E.3 Shell Command Language

41 *Editor's Note: The text of the Rationale for this section has been temporarily*
42 *located in Section 3, adjacent to the text it is explaining. The text will return to*
43 *this annex after the completion of balloting.*

44 **E.3.1 Shell Definitions**

45 **E.3.2 Quoting**

46 **E.3.3 Token Recognition**

47 **E.3.4 Reserved Words**

48 **E.3.5 Parameters and Variables**

49 **E.3.6 Word Expansions**

50 **E.3.7 Redirection**

51 **E.3.8 Exit Status for Commands**

52 **E.3.9 Shell Commands**

53 **E.3.10 Shell Grammar**

54 **E.3.11 Signals and Error Handling**

55 **E.3.12 Shell Execution Environment**

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

E.3.13 Pattern Matching Notation

E.3.14 Special Built-in Utilities

E.4 Execution Environment Utilities

Editor's Note: The text of the Rationale for this section has been temporarily located in Section 4, adjacent to the text it is explaining. The text will return to this annex after the completion of balloting. Notations regarding utilities probably included in the UPE have been updated, without diff marks, based on the current working draft of 1003.2a.

Many utilities were evaluated by the working group; more utilities were excluded from the standard than included. The following list contains many common UNIX system utilities that were not included as Execution Environment Utilities or in one of the Software Development Environment groups. It is logistically difficult for this Rationale to correctly distribute the reasons for not including a utility among the various utility environment sections. Therefore, this section covers the reasons for all utilities not included in Sections 4 and 6 and Annexes A and C.

The working group started its deliberations with a recommended list of utilities provided by the X/Open group of companies. This list was a subset of the utilities in the *X/Open Portability Guide*, *Issue II*, so it was very closely related to System V. The list had already been purged of purely administrative utilities, such as those found in System V's Administered System Extension. Then, the working group applied its scope as a filter and substantially pruned the remaining list as well.

The following list of ''rejected'' utilities is limited by its historical roots; since the selected utilities emerged from primarily a System V base, this list does not include sometimes familiar entries from BSD. The working group received substantial input from representatives of the University of California at Berkeley and from companies that are firmly allied with BSD versions of the UNIX system, enough so that some BSD-derived utilities are included in the standard. However, this Rationale is now limited to a discussion of only those utilities actively or indirectly evaluated by the working group, rather than the list of all known UNIX utilities from all its variants. This list will most likely be augmented during the balloting process as balloters request specific rationales for their favorite commands.

In the list, the notation [*POSIX .2a*] is used to identify utilities that are being evaluated for inclusion in the forthcoming User Portability Extension to this standard. Similarly, [*POSIX .7*] is used for those that may be appropriate for the working group evaluating system administration and [*POSIX .Net*] for networking standards.

`adb` The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This utility is primarily a debugging

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

tool. Furthermore, many useful aspects of `adb` are very hardware-specific.

`admin` The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This SCCS utility is primarily a development tool.

`as` Assemblers are hardware-specific and are included implicitly as part of the compilers in the standard.

`at` The `at` and `cron` family of utilities were omitted because portable applications could not rely on their behavior. [*POSIX .2a*]

`banner` The only known use of this command is as part of the LP printer header pages. It was decided that the format of the header is implementation defined, so this utility is superfluous to application portability.

`batch` The `at` and `cron` family of utilities were omitted because portable applications could not rely on their behavior. [*POSIX .2a*]

`cal` This calendar printing program is not useful to portable applications.

`calendar` This reminder service program is not useful to portable applications.

`cancel` The LP (line printer spooling) system specified is the most basic possible and did not need this level of application control. [*POSIX .7*]

`cflow` The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This utility is primarily a debugging tool.

`chroot` This is primarily of administrative use, requiring super-user privileges. [*POSIX .7*]

`col` No utilities defined in this standard produce output requiring such a filter. The `nroff` text formatter is present on many historical systems and will continue to remain as an extension; `col` is expected to be shipped by all the systems that ship `nroff`.

`cpio` This has been replaced by `pax`, for reasons explained in its own Rationale.

`cpp` Can be subsumed by `c89`.

`crontab` The `at` and `cron` family of utilities were omitted because portable applications could not rely on their behavior. [*POSIX .2a*]

`csplit` This utility's functionality can sometimes be provided by the `dd` or `sed` utilities (i.e., although these utilities cannot easily provide all of `csplit`'s features in one package, they can frequently be

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

used for the type of task that `csplit` is being used for). [*POSIX .2a*]

`cu` Terminal oriented—not useful from shell scripts or typical application programs. [*POSIX .Net*]

`cxref` The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This utility is primarily a debugging tool.

`dc` This utility's functionality can be provided by the `bc` utility; `bc` was selected because it was easier to use and had superior functionality. Although the historical versions of `bc` are implemented using `dc` as a base, this standard prescribes the interface and not the underlying mechanism used to implement it.

`delta` The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This SCCS utility is primarily a development tool.

`df` As the standard does not address the concept or nature of file systems, this command could not be specified in a manner useful to portable applications. [*POSIX .2a*]

`dircmp` Although a useful concept, the traditional output of this directory comparison program is not suitable for processing in applications programs. Also, the `diff -r` command gives equivalent functionality.

`dis` Disassemblers are hardware-specific.

`du` Because of differences between systems in measuring disk usage, this utility could not be used reliably by a portable application. [*POSIX .2a*]

`egrep` Marked obsolescent and replaced by the new version of `grep`.

`ex` This is typically a link to the `vi` terminal-oriented editor—not useful from shell scripts or typical application programs. The nonterminal oriented facilities of `ex` are provided by `ed`. [*POSIX .2a*]

`fgrep` Marked obsolescent and replaced by the new version of `grep`.

`file` Determining the type of file is generally accomplished with `test` or `find`. The added information available with `file` is of little use to a portable application, particularly since there is considerable variation in its output contents. [*POSIX .2a*]

`get` The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This SCCS utility is primarily a development tool.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

`ld` Is subsumed by `c89`.

`line` The functionality of `line` can be provided with `read`.

`lint` The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This utility is primarily a debugging tool.

`login` Terminal oriented—not useful from shell scripts or typical application programs.

`lorder` This utility is an aid in creating an implementation-specific detail of object libraries that the working group did not feel required standardization.

`lpstat` The LP system specified is the most basic possible and did not need this level of application control. [*POSIX .7*]

`m4` The working group did not find that this macro processor had sufficiently wide usage for standardization.

`mail` This utility was omitted in favor of `mailx`, because there was a considerable functionality overlap between the two. The mail-sending aspects of `mailx` are covered in this standard, the mail-reading in the UPE. [*POSIX .2a*]

`mesg` Terminal oriented—not useful from shell scripts or typical application programs. [*POSIX .2a*]

`mknod` This was omitted in favor of `mkfifo`, as `mknod` has too many implementation-defined functions. [*POSIX .7*]

`newgrp` Terminal oriented—not useful from shell scripts or typical application programs. [*POSIX .2a*]

`news` Terminal oriented—not useful from shell scripts or typical application programs.

`nice` Due to historical variations in usage, and in the lack of underlying support from possible POSIX.1 {8}base systems, this cannot be used by applications to achieve reliable results. [*POSIX .2a*]

`nl` The useful functionality of `nl` can be provided with `pr`.

`nm` The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This utility is primarily a debugging tool. [*POSIX .2a*]

`pack` The working group found little interest in a portable data compression program (and there are others that are probably more widely used anyway).

`passwd` Terminal oriented—not useful from shell scripts or typical application programs. (There was also sentiment to avoid security-

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

related utilities until requirements of 1003.6 are known.)

`pcat` The working group found little interest in a portable data compression program (and there are others that are probably more widely used anyway).

`pg` Terminal oriented—not useful from shell scripts or typical application programs.

`prof` The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This utility is primarily a debugging tool.

`prs` The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This SCCS utility is primarily a development tool.

`ps` This utility has historically been difficult to specify portably due to the many implementation-defined aspects of processes. Furthermore, a portable application can rarely rely on information about what other processes are doing, as security mechanisms may prevent it. A process requiring one of its children's process IDs (such as for use with the `kill` command) will have to record the IDs at the time of creation. [*POSIX .2a*]

`red` Restricted editor. This was not considered by the working group because it never provided the level of security restriction required.

`rmdel` The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This SCCS utility is primarily a development tool.

`rsh` Restricted shell. This was not considered by the working group because it does not provide the level of security restriction that is implied by historical documentation.

`sact` The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This SCCS utility is primarily a development tool.

`sdb` The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This utility is primarily a debugging tool. Furthermore, some useful aspects of `sdb` are very hardware-specific.

`sdiff` The ''side-by-side `diff`'' utility from System V was omitted because it is used infrequently, and even less so by portable applications. Despite being in System V, it is not in the *SVID* or *XPG*.

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

shar Utilities with this type of functionality (''shell-based archivers'') are in wide use, despite not being included in System V or BSD systems. However, the working group felt this sort of program was more widely used by human users than portable applications.

shl Terminal oriented—not useful from shell scripts or typical application programs. The job control aspects of the Shell Command Language are generally more useful and are being evaluated for the UPE.

size The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This utility is primarily a debugging tool.

spell Not useful from shell scripts or typical application programs.

split The functionality can sometimes be provided by the **dd**, **sed**, or (for some uses) **xargs** utilities (i.e., although these utilities cannot easily provide all of **split**'s features in one package, they can sometimes be used for the type of task that **split** is being used for). [*POSIX .2a*]

strings This is normally used by human users during debugging, rather than by applications. [*POSIX .2a*]

su Not useful from shell scripts or typical application programs. (There was also sentiment to avoid security-related utilities until requirements of POSIX. 6 are known.)

sum This utility was renamed **cksum**.

tabs Terminal oriented—not useful from shell scripts or typical application programs. [*POSIX .2a*]

time Not necessary for portable applications. It is frequently used by human users in debugging or for informal benchmarks. It is doubtful whether any standardized definitions of the output could be agreed upon.

tsort This utility is an aid in creating an implementation-specific detail of object libraries that the working group did not feel required standardization.

unget The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This SCCS utility is primarily a development tool.

unpack The working group found little interest in a portable data compression program (and there are others that are probably more widely used anyway).

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

`uucp`
`uulog`
`uupick`
`uustat`
`uuto` The UUCP utilities and their protocol description were removed from an early draft because responsibility for them was officially requested by the POSIX group developing networking interfaces.

`val` The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This SCCS utility is primarily a development tool.

`vi` Terminal oriented—not useful from shell scripts or typical application programs. [*POSIX .2a*]

`wall` Terminal oriented—not useful from shell scripts or typical application programs. It is generally used by system administrators, as well. [*POSIX .7*]

`what` The intent of the various software development utilities was to assist in the installation (rather than the actual development and debugging) of applications. This SCCS utility is primarily a development tool.

`who` The ability to determine other users on the system was felt to be at risk in a trusted implementation, so its use could not be considered by a portable application. [*POSIX .2a*]

`write` Terminal oriented—not useful from shell scripts or typical application programs. [*POSIX .2a*]

E.4.1 `awk` — Pattern scanning and processing language

E.4.2 `basename` — Return nondirectory portion of pathname

E.4.3 `bc` — Arbitrary-precision arithmetic language

E.4.4 `cat` — Concatenate and print files

E.4.5 `cd` — Change working directory

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

333 **E.4.6 `chgrp` — Change file group ownership**

334 **E.4.7 `chmod` — Change file modes**

335 **E.4.8 `chown` — Change file ownership**

336 **E.4.9 `cksum` — Write file checksums and block counts**

337 **E.4.10 `cmp` — Compare two files**

338 **E.4.11 `comm` — Select or reject lines common to two files**

339 **E.4.12 `command` — Select or reject lines common to two files**

340 **E.4.13 `cp` — Copy files**

341 **E.4.14 `cut` — Cut out selected fields of each line of a file**

342 **E.4.15 `date` — Write the date and time**

343 **E.4.16 `dd` — Convert and copy a file**

344 **E.4.17 `diff` — Compare two files**

345 **E.4.18 `dirname` — Return directory portion of pathname**

346 **E.4.19 `echo` — Write arguments to standard output**

347 **E.4.20 `ed` — Edit text**

348 **E.4.21 `env` — Set environment for command invocation**

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.

E.4.22 `expr` — Evaluate arguments as an expression

E.4.23 `false` — Return false value

E.4.24 `find` — Find files

E.4.25 `fold` — Filter for folding lines

E.4.26 `getconf` — Get configuration values

E.4.27 `getopts` — Parse utility options

E.4.28 `grep` — File pattern searcher

E.4.29 `head` — Copy the first part of files

E.4.30 `id` — Return user identity

E.4.31 `join` — Relational database operator

E.4.32 `kill` — Terminate or signal processes

E.4.33 `ln` — Link files

E.4.34 `locale` — Get locale-specific information

E.4.35 `localedef` — Define locale environment

E.4.36 `logger` — Log messages

E.4.37 `logname` — Return user's login name

Copyright © 1991 IEEE. All rights reserved.
This is an unapproved IEEE Standards Draft, subject to change.