

IEEE Standards Interpretations for IEEE Std 1003.2-1992

Circuits and Devices

Communications Technology

IEEE Computer Society

Sponsored by the
Portable Applications Standards Committee

*Electromagnetics and
Radiation*

Energy and Power

Industrial Applications

*Signals and
Applications*

*Standards
Coordinating
Committees*



Published by the Institute of Electrical and Electronics Engineers, Inc., 345 East 47th Street, New York, NY 10017, USA.

December 30, 1994

SH94253

IEEE Standards Interpretations

for IEEE Std 1003.2-1992

Sponsor

**Portable Applications Standards Committee
of the
IEEE Computer Society**



Abstract: The Portable Applications Standards Committee of the IEEE Computer Society carried out a series of analyses of various problems encountered by users of IEEE Std 1003.2-1992, IEEE Standard for Information Technology—Portable Operating System Interface (POSIX)—Part 2: Shell and Utilities. The results of its deliberations are presented in this document. The intent is to give the POSIX community reasonable ways of interpreting unclear portions of the standard.

Keywords: computer, computer environments, interfaces, interpretations, portable operating system interface, POSIX, shell, utilities

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

Copyright © 1994 by the Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 1994. Printed in the United States of America.

ISBN 1-55937-476-4

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 Interpretations

Occasionally, questions may arise regarding the meaning of portions of standards as they relate to specific applications. Such requests for interpretations should ask for clarifications of the exact nature of the contents of the standard.

Questions relating to such interpretations are reviewed and evaluated by a balance of members representing the specific committee interests. This officially formed interpretations subgroup creates and circulates a draft interpretation among its members. It transmits a final interpretation to the party initiating the request only after consensus has been achieved. The interpretation is also given to the standards-developing committee to consider addressing it in a supplement or the next revision to the standard.

Interpretations are issued to explain and clarify the intent of the standard and are not intended to constitute an alteration to the original standard or to supply consulting information. The interpretations subgroup cannot make new rules to fit situations not yet covered in the standard, even if the investigations of the subgroup lead it to conclude that the requirement is incomplete or in error. Changes to the standard are made only through revisions or supplements to the standard.

It is recognized that requests are frequently received that are partially or totally requests for information rather than requests for an interpretation. It is inappropriate to issue an official interpretation to answer such requests. The interpretations subgroup may, however, find from its research that the literal printing of the standards text is not identical to that approved by the standards developers and may issue an editorial correction as a part of its interpretation.

The original interpretations requests in this document have been lightly edited to remove extraneous matter and to focus on the problem presented. Some illustrations have been redrawn for publication. With these exceptions, requests are in the form received.

Procedure for Requesting an Interpretation

Requests for interpretations should be addressed to

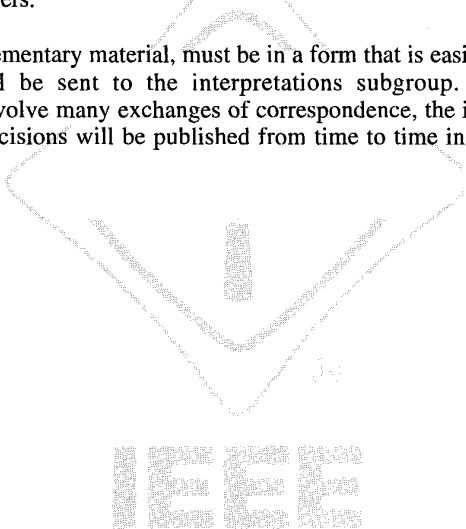
Secretary, IEEE Standards Board
IEEE Standards Department
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331

Requests for interpretations should include

- a) The specific designation of the standard, including the year of publication.
- b) The specific subsection being questioned.
- c) The applicable conditions for the case in question.

Line drawings should be black ink or excellent black pencil originals. Photos should be black-and-white glossy prints. These illustrations must be reproduced for committee circulation and eventually will be used to supplement the text of the next edition. Clear diagrams and pictures will make the work of interpretation easier and more valuable to users.

Requests, including all supplementary material, must be in a form that is easily reproducible. If suitable for consideration, requests will be sent to the interpretations subgroup. After consideration by the subcommittee, which may involve many exchanges of correspondence, the inquirer will be notified of the decision of the subgroup. Decisions will be published from time to time in cumulative form and may be ordered from the IEEE.



Introduction

After approval and publication of a standard, questions may arise regarding the meaning of portions of a standards as it relates to its users. In the case of IEEE Std 1003.2-1992, these questions arose from both system implementors and application authors. When a need for interpretation is brought to the attention of the IEEE, the Institute initiates action to prepare appropriate responses.

IEEE Std 1003.2-1992 (POSIX.2) was developed by the Portable Applications Standards Committee (PASC) of the IEEE Computer Society. Each interpretation request is forwarded to the members of the PASC interpretations group and a subgroup is invited to generate a draft interpretation for review. The draft interpretation is made available to the whole interpretations group for review and comment. When consensus has been reached within the interpretations group, the IEEE transmits the final interpretation to the party initiating the request.

It is recognized that requests are frequently received that are partially or totally requests for information rather than requests for interpretation. The IEEE considers it inappropriate to issue an official interpretation to answer such requests. This includes requests for advice as to the means by which particular facilities described in the standard could be implemented.

Every IEEE standard is subjected to review at least every five years for revision or reaffirmation. The revision process takes input from a wide variety of sources in order to develop a revised standard. One of these sources is the interpretations that have been issued against the extant version of the standard. It is not to be expected that the text of the interpretation will appear in the revised standard, but rather that the revision will address the issues raised by the interpretation request through the consensus process. The revision process is allowed to resolve ambiguities in the original standard through textual change; this capability is not allowed within the interpretations process.

This document contains a compilation of all of the interpretations completed against IEEE Std 1003.2-1992 as of November 1994. The IEEE will provide details of interpretation requests received that have not yet appeared in a publication. Where a formal interpretation has been completed, this will include the text of the interpretation.

At the time that this compilation of interpretations was published, the following members had participated in the IEEE Std 1003.2-1992 interpretations group:

Portable Applications Standards Committee

Chair:	Lowell Johnson
Vice-Chair, Interpretations:	Andrew Josey

Keith Bostic	Eric Horner	David Rowley
Mark Brown	Hal Jespersen	Thomas Shem
Dawn Burnett	William King	Scott Sutter
Don Cragun	Bob Korn	Marc Teitelbaum
Dave Decot	David Korn	Alex White
John Farley	Shane McCarron	David Willcox
Mark Funkenhauser	Gary Miller	David Williams
Jeff Haemer	Martha Nalebuff	Jeff Zado

Rachel A. Meisel
IEEE Standards Project Editor

Interpretation Number: 3 1
Topic: ex—Replacement strings
Relevant Clauses: 5.10.7.4
Classification: Defect

Interpretation request

Historically, the upper and lower case manipulations occurred on all characters in the replacement strings, not just those inserted using \n or \[1-9]. Was it the intent of the standard to change historical practice? If not, the following wording is suggested:

The strings \l, \u, \L, and \U can be used to modify the case of elements in the replacement string. The string \l (\u) shall cause the character that follows to be converted to lowercase (uppercase). The strings \L (\U) shall cause all characters subsequent to them to be converted to lowercase (uppercase) until the string \e or \E, or the end of the replacement string, is encountered.

Interpretation for IEEE Std 1003.2-1992

The standard is specific in its requirement that \l, \u, \L, and \U should only modify the case of elements using the & and \digit notation.

Concerns about the wording of this part of the standard have been forwarded to the sponsor.

Rationale for interpretation

None.

Editorial note for future revision of standard (not part of this interpretation)

Consider applying the above proposed change to 5.10.7.4, Replacement Strings.

NOTE—There is no certainty that these comments will apply to any revision of IEEE Std 1003.2-1992.

Interpretation Number: 32
Topic: ed—List command
Relevant Clauses: 4.20.7.3.14
Classification: Defect

Interpretation request

The newline (`\n`) character is listed in Table 2-16. Historically, the *list* command within *ed* did not display the newline terminating each line in a special mode. Was it the intent of the standard to change this?

Interpretation for IEEE Std 1003.2-1992

The standard states that *ed* should display the newline terminating each line in a special mode for the *list* command and conforming implementations shall conform to this.

Concerns about the wording of this part of the standard have been forwarded to the sponsor.

Rationale for interpretation

None.

Editorial note for future revision of standard (not part of this interpretation)

A proposed change to 4.20.7.3.14, List Command, is to replace the second sentence, starting on line 3825, with the following:

The characters listed in Table 2-16 (see 2.12), except for `\n`, shall be written as the corresponding escape sequences.

NOTE—There is no certainty that these comments will apply to any revision of IEEE Std 1003.2-1992.

Interpretation Number: 33
Topic: sed—Extended description
Relevant Clauses: 4.55.7
Classification: Ambiguous

Interpretation request

Was it the intent of the standard to use *command* both in line 8629 and again in the subsequent text (particularly in line 8803)? The two uses were not the same in historic implementations of *sed*.

Interpretation for IEEE Std 1003.2-1992

The first part of 4.55.7 describes the form of a command using the term *command*, thereby making the meaning of *command* in 4.55.7 ambiguous.

The standard is unclear on this issue, and no conformance distinction can be made between alternative implementations based on this. This is being referred to the sponsor.

Rationale for interpretation

None.



Interpretation Number: 34
Topic: sed REs
Relevant Clauses: 4.55.7.2
Classification: Unaddressed issue

Interpretation request

Was it the intent of the standard to ignore empty REs? Historic implementations of sed used the empty RE (//) to match the last RE used.

Interpretation for IEEE Std 1003.2-1992

The meaning of an empty RE is not defined by the standard.

The standard does not speak to the meaning of a null BRE for sed. The standard is unclear on this issue, and no conformance distinction can be made between alternative implementations based on this. This is being referred to the sponsor.

Rationale for interpretation

None.

Editorial note for future revision of standard (not part of this interpretation)

A proposed change to 4.55.7.2, sed REs, is to add the following paragraph (3):

- (3) If an RE is empty (i.e., no pattern is specified) sed shall behave as if the last RE used in the last command applied (either as an address or as part of a substitute command) was specified.

NOTE—There is no certainty that these comments will apply to any revision of IEEE Std 1003.2-1992.

Interpretation Number: 35.1
Topic: sed editing commands
Relevant Clauses: 4.55.7.3
Classification: Unaddressed issues

Interpretation request

- (1) Was it the intent of the standard to not describe that multiple commands may be entered on a single line by interspersing the commands with semi-colons?
- (2) Page 412, lines 8784–8785: Does the standard require white space before the *wfile* argument to the *w* flag to the substitute command?
- (3) Page 413, lines 8794–8802: Was it the intent of the standard to ignore that the *y* command has historically used the string `\n` to cause *sed* to insert a newline into the string?
- (4) Page 413, lines 8803–8806: It is unclear in the standard what happens when the *command* is itself the `!` character, i.e., what if the user enters

[2addr]!!!!

Interpretation for IEEE Std 1003.2-1992

- (1) The purpose of the interpretation process is not to determine intent. Using a semi-colon character (`;`) in *sed* to separate commands would be a legitimate extension. However, in a few cases, (for example, the *w* command) the standard would require that the semi-colon character be interpreted as part of *wfile* unless there was white space between *wfile* and the semi-colon character.

The standard is unclear on these issues, and no conformance distinction can be made between alternative implementations based on this. This is being referred to the sponsor.

- (2) The standard is unclear on these issues, and no conformance distinction can be made between alternative implementations based on this. This is being referred to the sponsor.
- (3) The standard is unclear on these issues, and no conformance distinction can be made between alternative implementations based on this. This is being referred to the sponsor.
- (4) The behavior of [2addr]!!!! is unspecified. The standard is unclear on these issues, and no conformance distinction can be made between alternative implementations based on this. This is being referred to the sponsor.

Rationale for interpretation

None.

Interpretation Number: 35.2
Topic: sed editing commands
Relevant Clauses: 4.55.7.3
Classification: Defects

Interpretation request

- (1) Page 410, lines 8700–8703: Was it the intention of the standard to change historical practice by not including `n` as a command that can cause the appended text to be written to standard output?
- (2) Page 411, lines 8727–8734: The newline character is listed in Table 2-16. Was it the intent of the standard to change historical practice by having `sed` display the newline terminating each line in a special format?
- (3) Page 412, lines 8767–8770: Was it the intent of the standard to change historic practice by not discarding the backslashes from the replacement string?

Interpretation for IEEE Std 1003.2-1992

- (1) The standard states that the `N` command only is acceptable in this case, and conforming implementations shall conform to this.

Concerns about the wording of this part of the standard have been forwarded to the sponsor.

- (2) The standard states that `sed` shall display the newline terminating each line in a special format, and conforming implementations shall conform to this.

Concerns about the wording of this part of the standard have been forwarded to the sponsor.

- (3) The standard states on page 412, line 8767, that

“For each backslash (`\`) encountered in scanning *replacement* from beginning to end, the following character shall lose its special meaning (if any).”

and conforming implementations shall conform to this.

Concerns about the wording of this part of the standard have been forwarded to the sponsor.

Rationale for interpretation

None.

Interpretation Number: 35.3
Topic: sed editing commands
Relevant Clauses: 4.55.7.3
Classification: Ambiguous

Interpretation request

On pages 411, lines 8739–8744, was it the intent of the standard to change historical practice by not having the `n` and `N` commands branch to the end of the script if they encounter the end of the input?

Interpretation for IEEE Std 1003.2-1992

The standard is unclear regarding the specific behavior of the `sed` commands `n` and `N` when the end of input is reached while in a script, and no conformance distinction can be made between alternative implementations based on this. This is being referred to the sponsor.

Rationale for interpretation

None.



Interpretation Number: 36
Topic: ed commands and extended description
Relevant Clauses: 4.20.7.3, 5.10.7, 5.35.7
Classification: Ambiguous

Interpretation request

The standard does not seem to describe what behavior is required from the three editors `ed`, `ex`, and `vi` when an end-of-file is encountered during either command input or text input. Please clarify what the utility implementations are expected to do.

Historic practice appears to be both inconsistent and wrong. It is strongly believed that the commands should behave very similarly to what happens when a `SIGHUP` is received, i.e., save the user's work so that it can be recovered, and exit with an error status.

Interpretation for IEEE Std 1003.2-1992

The standard is unclear on this issue, and no conformance distinction can be made between alternative implementations based on this. This is being referred to the sponsor.

Rationale for interpretation

None.

Editorial note for future revision of standard (not part of this interpretation)

A proposed change to 4.20.7.3, `ed` Commands, is to replace the sixth paragraph (lines 3690–3691) with the following:

If an end-of-file is detected on standard input when input from the user is expected and the standard input is a terminal device, if the buffer is not empty and has changed since the last write, the `ed` utility shall attempt to write a copy of the buffer to a file. First, the file named `ed.hup` in the current directory shall be used; if that fails, the file named `ed.hup` in the directory named by the **HOME** environment variable shall be used.

If an end-of-file is detected on standard input when input from the user is expected and standard input is not a terminal device, the `ed` utility shall not write the file to the currently remembered pathname or return to command mode, and shall terminate with a nonzero exit status.

NOTE—There is no certainty that these comments will apply to any revision of IEEE Std 1003.2-1992.

Interpretation Number: 37
Topic: ~~more~~—Input files
Relevant Clauses: 5.18.5.2
Classification: Defect

Interpretation request

Was it the intent of the standard to disallow the historic practice of using `/dev/tty` as an input device if standard error is not readable?

Interpretation for IEEE Std 1003.2-1992

The standard states on lines 2807–2810 that if standard output is a terminal, standard error is not readable, and command input is needed and ~~more~~ shall terminate with an error indicating that it was unable to read user commands. Conforming implementations shall conform to this.

Concerns about the wording of this part of the standard have been forwarded to the sponsor.

Rationale for interpretation

None.

Editorial note for future revision of standard (not part of this interpretation)

A proposed change to 5.18.5.2, Input Files, is to replace the text of the subclause with the following:

The input files being examined shall be text files. If the standard output is a terminal, standard error shall be used to read commands from the user. If the standard output is a terminal, standard error is not readable, and command input is needed, ~~more~~ may attempt to obtain user commands from the controlling terminal (e.g., `/dev/tty`); otherwise, ~~more~~ shall terminate with an error indicating it was unable to read user commands. If standard output is not a terminal, no error shall result if standard error cannot be opened for reading.

NOTE—There is no certainty that these comments will apply to any revision of IEEE Std 1003.2-1992.

Interpretation Number: 38
Topic: ex—Extended description
Relevant Clauses: 5.10.7
Classification: Defect

Interpretation request

The specification that “no .exrc file shall be read unless it is owned by the same user ID as the effective user ID of the process” is necessary but not sufficient. To keep the .exrc files from being a security problem, the file should not be read if it is writable by anyone other than the owner.

Interpretation for IEEE Std 1003.2-1992

The standard states the required behavior and conforming implementations shall conform to this.

Concerns about the wording of this part of the standard have been forwarded to the sponsor.

Rationale for interpretation

Interpretations cannot make substantive changes to the standard. This may be considered for a future revision.

Interpretation Number: 39
Topic: `ls`
Relevant Clauses: 4.39.8
Classification: No change

Interpretation request

Line 6071 states that `ls` should exit with 0 if “all files were written successfully.” The `ls` command does not normally write files, but this implies that it does.

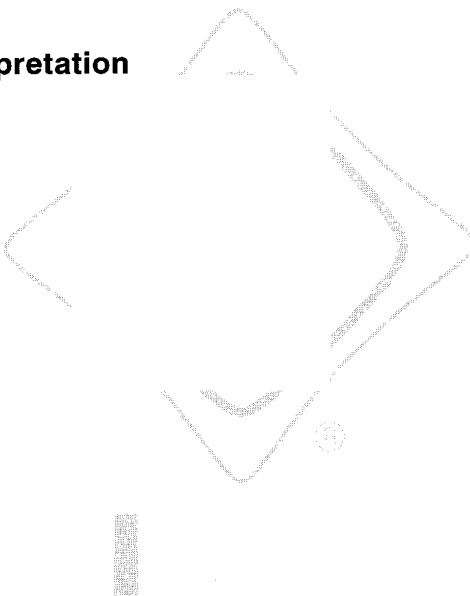
A clarification of the intended behavior is required.

Interpretation for IEEE Std 1003.2-1992

The sentence means the names of all relevant files were written successfully.

Rationale for interpretation

None.



Interpretation Number: 40
Topic: Internationalization issues
Relevant Clauses: 2.5.2.2
Classification: Ambiguous

Interpretation request

POSIX.2 has defined a mechanism for talking about multi-character sequences as a single unit, namely as collating elements (CEs). Although CEs are motivated by sorting issues, they appear in REs. This leads to the question of how to parse text into CEs. There are many possible answers, and furthermore, the parsing might be affected by context. For example, given the usual alphabet augmented by the collating element `<ij>` defined as `<i><j>`, can the string `ij` ever be parsed as two collating elements?

Subclause 2.5.2.2, line 1668, says in the context of sorting, “strings are first broken up into a series of collating elements.” Does this apply to pattern matching? And if so, how exactly is this done (for sorting or pattern matching)?

A proposed solution is to add the following text somewhere within 2.5.2.2. This text should be referred to by line 1668 and by the general RE introduction (2.8.2).

When a string is interpreted as a sequence of CEs, the sequence shall be as found by the following process:

Starting at the first character of the string, determine the longest prefix of the string that matches a CE, add that CE to the sequence and continue this process with the character after that prefix until the string is exhausted.

Note that this applies even if a sort key indicates that a piece of the text is processed in backwards (right-to-left) order; that is, the right-to-left processing applies to the CEs found by a left-to-right lexical scan.

Rationale for interpretation

This is the greedy algorithm normally done in lexical analysis. Any other choice would require backtracking with potentially exponential runtime. It implies that, when `<i><j>` is a collating element, under no circumstances can a bracket expression match the `i` alone in the string `ij`. In particular, neither `[ [.i.] [.ij.] ] j` nor `[ [.i.] ] j` matches `ij`. By contrast, `i [ [.j.] ]` does match `ij`, because in this regular expression `i` denotes a character and is unaffected by concerns about collating elements.

Interpretation for IEEE Std 1003.2-1992

The standard is unclear on this issue, and no conformance distinction can be made between alternative implementations based on this. This is being referred to the sponsor.

Rationale for interpretation

None.

Interpretation Number: 4 1
Topic: Internationalization issues—Locales
Relevant Clauses: 2.5.2.1, 2.8.3.2 (6)
Classification: 1–6: Unaddressed issues
7: Ambiguous issue
8: No change

Interpretation request

- (1) The specification of locales and the interface to them discriminates against nonvendor-supplied software. In particular, it is impossible to write a portable implementation of *regcomp()* and *regexexec()*, as there is no standardized interface to the vital knowledge presumably set up by a call to *setlocale()*. This knowledge is detailed below. In brief, the first seems an oversight and the others are necessary to use the locale information.

- (2) How can membership in class `[:blank:]` be determined portably?

A proposed solution is to provide a *ctype* function *isblank()*.

Rationale—It is inconsistent that this be the only `LC_CTYPE` category without a C binding. Note that this extension introduces a difference between the C and POSIX locales.

- (3) How can the meaning of an arbitrary equivalence class be discovered portably?

A proposed solution is to provide a function that, given any name for an equivalence class, returns a list of names of collating symbols in the class. The order of the list shall be the same regardless of what name is given.

Rationale—This is needed if an application, such as a searching or sorting tool, requires this locale-specific information. In particular *regcomp()* and *sort* need it.

- (4) How can the meaning/value of an arbitrary collating symbol be determined portably?

A proposed solution is to provide a function that, given a collating symbol, returns the representation and length of the symbol.

Rationale—This is needed if an application, such as a searching or sorting tool, requires this locale-specific information.

- (5) How can the collating elements in a string be found and compared portably?

A proposed solution is to provide a function that returns the length and the weight vector for the collating element at the beginning of the string.

Rationale—This is needed if an application, such as a searching or sorting tool, requires this locale-specific information.

- (6) How can *regcomp()* expand a range expression into a list of collating elements portably?

A proposed solution is to provide a successor function that, given the name of a collating element, returns the name of the collating element with the next larger weight vector. For this purpose two elements with the same weight vector compare in the order of their equivalence listing.

Rationale—This is needed if an application, such as a searching or sorting tool, requires this locale-specific information. It may further be useful to have a way to inquire whether a locale contains any multi-character collating elements.

- (7) Lines 2918–2920 say that an equivalence class expression that names a collating element not in an equivalence class shall be treated as a collating symbol. Does this statement affect the meaning of `collating_symbol` in line 3306? Does it eliminate such equivalence class expressions from consideration in lines 2943–2945?

A proposed solution is to change lines 2918–2920 to say “the expression shall be understood as an equivalence class that contains only the one collating element.”

Actually, the admittance of singleton equivalence classes in the definitions of 2.5.2.2 would be preferred.

Rationale—This question affects the meaning of range expressions. Lines 2918–2920 could be construed as forcing `[ [=CE1=] - [=CE2=] ]` to mean `[ [.CE1.] - [.CE2.] ]` in some cases, although the former expression looks syntactically incorrect. The preferred solution agrees with customary mathematical usage, and clarifies the behavior of the equivalence-class function proposed in (3) above.

- (8) What if collation changes between `regcomp()` and `regexexec()`?

A proposed solution is that the result is undefined.

Rationale—For the common case of locales in which all collating elements are single characters, `regcomp()` should be allowed to compile character classes. At the same time, `regexexec()` should be allowed to handle multi-character collating symbols. The proposed resolution assures that both desiderata are met.

Interpretation for IEEE Std 1003.2-1992

- (1) The standard does not speak to this issue and no conformance distinction can be made between alternative implementations based on this.

The standard does not require that an implementation conforming to the standard be portable. Therefore, there is no requirement that the functionality be specified by the standard.

Concerns are being forwarded to the sponsor.

- (2–6) The standard does not speak to these issues and no conformance distinction can be made between alternative implementations based on this.

Concerns are being forwarded to the sponsor.

- (7) The standard is unclear on this issue, and as such no conformance distinction can be made between alternative implementations based on this. This is being referred to the sponsor.

- (8) The standard states the required behavior and conforming implementations shall conform to this. According to B.5.2, lines 367–368 on page 729, the standard specifies the result is undefined.

Rationale for interpretation

None.