

## IEEE Standards Interpretations for IEEE Std 1003.2™-1992 IEEE Standard for Information Technology--Portable Operating System Interfaces (POSIX®)-- Part 2: Shell and Utilities

Copyright © 1996 by the Institute of Electrical and Electronics Engineers, Inc. 3 Park Avenue New York, New York 10016-5997 USA All Rights Reserved.

Interpretations are issued to explain and clarify the intent of a standard and **do not** constitute an alteration to the original standard. In addition, interpretations are not intended to supply consulting information. Permission is hereby granted to download and print one copy of this document. Individuals seeking permission to reproduce and/or distribute this document in its entirety or portions of this document must contact the IEEE Standards Department for the appropriate license. Use of the information contained in this document is at your own risk.

IEEE Standards Department, Copyrights and Permissions, 445 Hoes Lane, Piscataway, New Jersey 08855-1331, USA

### Interpretation Request #91

**Topic:** awk - gsub/sub **Relevant Clauses:** 4.1.7.6.2.2

In subclause 4.1.7.6.2.2, page 178, lines 647-653, the text for gsub() and sub() concerning the use of backslash states: For each occurrence of backslash (\) encountered when scanning the string `_repl_` from beginning to end, the next character shall be taken literally and lose its special meaning (e.g. `&` shall be interpreted as a literal ampersand character). Except for `&` and `\`, it is unspecified what the special meaning of any such character is. This text implies that the only portable way to write the string `_repl_` is to put a backslash in front of every literal character, since there is no way to tell what characters may be special for any particular implementation of awk. This wording also does not seem to allow historical behaviour. Historically, awk treated the backslash character as an escape character and allowed the characters such as `"\b"` or `"\t"` to indicate a backspace and tab respectively.

The historical behaviour can be describe as: In `_repl_`, all characters are treated as literals, except for ampersand (`&`) and backslash (`\`). An ampersand (`&`) appearing in the string `_repl_` shall be replaced by the string from `_in_` that matches the ERE. Backslashes (`\`) in `_repl_` introduce an escape sequence. The sequence `\&` is replaced by a literal ampersand and `\\` is replaced by a literal backslash. The behaviour of a backslash followed by any other character is unspecified. Can you please provide Rationale as to why this non-historical behaviour was documented in 1003.2- 1992. Or provide an interpretation that allows a conforming implementation to provide the historical behaviour. If not, then this matter should be forwarded to the sponsors.

### Interpretation Response

The standard states behavior for the backslash (`\`) character, and conforming implemen-

tations must conform to this. However, concerns have been raised about this which are being referred to the sponsor.

**Rationale for Interpretation**

None.