INTERNATIONAL STANDARD ISO/IEC 9945-4:2003
TECHNICAL CORRIGENDUM 1

Published 2004-09-15

Information technology — Portable Operating System Interface (POSIX®) —

Part 4:
Rationale

TECHNICAL CORRIGENDUM 1

Technologies de l'information — Interface pour la portabilité des systèmes (POSIX®) —

Partie 4: Rationnel

RECTIFICATIF TECHNIQUE 1

Technical Corrigendum 1 to ISO/IEC 9945-4:2003 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 22, *Programming languages, their environments and system software interfaces.*

ICS 35.060 **Ref. No. ISO/IEC 9945-4:2003/Cor.1:2004(E)**

1 Scope

This technical corrigendum addresses issues raised in defect reports and interpretation requests submitted up to 14 August 2003, and that meet all of the following criteria:
a. They are in the scope of the approved International Standard.
b. They contain no new APIs (functions/utilities), however, they may add enumeration symbols, non-function # defines, and reserve additional namespaces.
c. They address contradictions between different parts of the International Standard, or add consistency between it and overriding International Standards, or address security-related problems.

2 Changes to ISO/IEC 9945-4

Change Number: XRAT/TC1/1 [XBD ERN 20]
On Page: xxxiii Line: "ISO/IEC 8859" Section: Referenced Documents
Add after line starting "Part 10":
"Part 11: Latin/Thai Alphabet"
Add after line starting "Part 15":
"Part 16: Latin Alphabet No. 10"

Change Number: XRAT/TC1/2 [XBD ERN 35]
On Page: 7 Line: 191 Section: A.1.4
Add to the end of the paragaph:
"In some places the text refers to facilities supplied by the implementation that are outside of the standard as implementation-supplied or implementation-provided. This is not intended to imply a requirement for documentation. If it were the term "implementation-defined" would have been used."

Change Number: XRAT/TC1/3 [XBD ERN 17]
On Page: 42 Line: 1652 Section: A.6.3
Change From:
"There is no additional rationale provided for this section."
To:
"The standard does not specify how wide characters are encoded or provide a method for defining wide characters in a charmap. It specifies ways of translating between wide characters and multibyte characters. The standard does not prevent an extension from providing a method to define wide characters."

Change Number: XRAT/TC1/4 [XBD ERN 25]
On Page: 49 Line: 1962-1966 Section: A.7.3.3
Change From:
"The locale definition is an extension of the ISO C standard localeconv() specification. In particular, rules on how currency_symbol is treated are extended to also cover int_curr_symbol, and p_set_by_space and n_sep_by_space have been augmented with the value 2, which places a <space> between the sign and the symbol (if they are adjacent; otherwise, it should be treated as a 0). The following table shows the result of various combinations:"
To:
"The locale definition is an extension of the ISO C standard localeconv() specification. In particular, rules on how currency_symbol is treated are extended to also cover int_curr_symbol, and p_set_by_space and n_sep_by_space have been augmented with the value 2, which places a <space> between the sign and the symbol. This has been updated to match the C99 requirements and is an incompatible change from UNIX 98 and POSIX.2-1992 and POSIX.1-1996 requirements. The following table shows the result of various combinations:"

Change Number: XRAT/TC1/5 [XSH ERN 92]
On Page: 55 Line: 2254-2257,2258-2259 Section: 8.2
In Section 8.2 Internationalization Variables
Change From:
"The text about locale implies that any utilities written in standard C and conforming to IEEE Std 1003.1-2001 must issue the following call:
```
setlocale(LC_ALL, "")
```

If this were omitted, the ISO C standard specifies that the C locale would be used."
To:
"Utilities conforming to the Shell and Utilities volume of IEEE Std 1003.1-2001 and written in standard C can access the locale variables by issuing the following call:

```
 setlocale(LC_ALL, "")
```

If this were omitted, the ISO C standard specifies that the C locale would be used."
Change From:
"If any of the environment variables are invalid, it makes sense to default to an implementation-defined, consistent locale environment."
To:
"The DESCRIPTION of setlocale() requires that when setting all categories of a locale,that if the value of any of the environment-variable searches yields a locale that is not supported (and nonnull), the setlocale() function returns a null pointer and the locale of the process is unchanged.
For the standard utilities, if any of the environment variables are invalid, it makes sense to default to an implementation-defined, consistent locale environment."

Change Number: XRAT/TC1/6 [XRAT ERN 1]
On Page: 72 Line: 2960-2961 Section: A.12.2
Change From:
"Guidelines 1 and 2 are offered as guidance for locales using Latin alphabets. No recommendations are made by IEEE Std 1003.1-2001 concerning utility naming in other locales."
To:
"Guidelines 1 and 2 encourage utility writers to use only characters from the portable character set because use of locale specific characters may make the utility inaccessible from other locales. Use of uppercase letters is discouraged due to problems associated with porting utilities to systems that don't distinguish between uppercase and lowercase characters in file names. Use of non-alphanumeric characters is discouraged due to the number of utilities that treat non-alphanumeric characters in "special" ways depending on context (such as the shell using whitespace characters to delimit arguments, various quote characters for quoting, <dollar-sign> to introduce variable expansion,...)"

Change Number: XRAT/TC1/7 [XSH ERN 89]
On Page: 177 Line: 7479 Section: B.2 Threads
Insert new section before B.2.10:
"B.2.9.8 Use of Application-Managed Thread Stacks
IEEE Std 1003.1-2001/Cor 2-200x, item XSH/TC1/8 is applied, adding this new section. It was added to make it clear that the current standard does not allow an application to determine when a stack can be reclaimed. This may be addressed in a future revision."

Change Number: XRAT/TC1/8 [XBD ERN 11]
On Page: 224 Line: 9264-9265 Section: C.1.9
Delete the paragraph:
" {POSIX2_SYMLINKS} was developed even though there is no comparable configuration value for the system interfaces."
Rationale: The pathconf() symbolic constant _PC_2_SYMLINKS has been introduced as a result of this defect report.

Change Number: XRAT/TC1/9 [XRAT ERN 2]
On Page: 240,241 Line: 9903-9911,9921,9925,9958 Section: C.2.6.4
Change From:
"The "(())" form of KornShell arithmetic in early proposals was omitted. The standard developers concluded that there was a strong desire for some kind of arithmetic evaluator to replace expr, and that relating it to '$' makes it work well with the standard shell language, and it provides access to arithmetic evaluation in places where accessing a utility would be inconvenient.
The syntax and semantics for arithmetic were changed for the ISO/IEC 9945-2:1993 standard. The language is essentially a pure arithmetic evaluator of constants and operators (excluding assignment) and represents a simple subset of the previous arithmetic language (which was derived from the KornShell "(())" construct). "

To:
"The standard developers agreed that there was a strong desire for some kind of arithmetic evaluator to provide functionality similar to expr, that relating it to '$' makes it work well with the standard shell language and provides access to arithmetic evaluation in places where accessing a utility would be inconvenient.
The syntax and semantics for arithmetic were revised for the ISO/IEC 9945-2:1993 standard. The language represents a simple subset of the previous arithmetic language (which was derived from the KornShell "(())" construct). "
On line 9921, add before the paragraph commencing "In early proposals"
"The standard requires assignment operators to be supported (as listed in Section 2.7.1), and since arithmetic expansions are not specified to be evaluated in a subshell environment, changes to variables there have to be in effect after the arithmetic expansion, just as in the parameter expansion ${x=value}.
Note, however, that $((x=5)) need not be equivalent to $(($x=5)).
If the value of the environment variable x is the string "y=", the expansion of $((x=5)) would set x to 5 and output 5, but $(($x=5)) output 0 if the value of the environment variable y is not 5 and would output 1 if the environment variable y is 5. Similarly, if the value of the environment variable is 4, the expansion of $((x=5)) would still set x to 5 and output 5, but $(($x=5)) (which would be equivalent to $((4=5))) would yield a syntax error. "
In the paragraph on line 9925
Change From:
"The portion of the ISO C standard arithmetic operations selected corresponds to the operations historically supported in the KornShell."
To:
"The portion of the ISO C standard arithmetic operations selected corresponds to the operations historically supported in the KornShell. In addition to the exceptions listed in section 2.6.4, the use of the following are explicitly outside the scope of the rules defined in section 1.7.2.1:
* The prefix operator & and the [], -> and . operators.
* Casts."
On line 9958, add before the paragraph commencing: "Although the ISO/IEC 9899:1999 Standard now..." :
"The standard is intentionally silent about how a variable's numeric value in an expression is determined from its normal "sequence of bytes" value. It could be done as a text substitution, as a conversion like that performed by strtol(), or even recursive evaluation. Therefore, the only cases for which the standard is clear are those for which both conversions produce the same result. The cases where they give the same result are those where the sequence of bytes form a valid integer constant. Therefore, if a variable does not contain a valid integer constant, the behavior is unspecified.
For the commands:

```
  x=010; echo $((x += 1))
```

the output must be 9.
For the commands:

```
 x=' 1'; echo $((x += 1))
```

the results are unspecified.
For the commands:

```
 x=1+1; echo $((x += 1))
```

the results are unspecified."