OS/390

OpenEdition
XPG4 Conformance Document

OS/390

OpenEdition
XPG4 Conformance Document

Note

Before using this information and the product it supports, be sure to read the general information under “Notices” on page vii.

| **Fifth Edition (September 1997)**

| This edition applies to Release 4 of OS/390 (5645-001) and to all subsequent releases and modifications until otherwise indicated in
| new editions.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address below.

IBM welcomes your comments. A form for readers' comments may be provided at the back of this publication, or you may address your comments to the following address:

International Business Machines Corporation
Department 55JA, Mail Station P384
522 South Road
Poughkeepsie, NY 12601-5400
United States of America

FAX (United States & Canada): 1+914+432-9405
FAX (Other Countries):
Your International Access Code +1+914+432-9405

IBMLink (United States customers only): KGNVMC(MHVRCFS)
IBM Mail Exchange: USIB6TC9 at IBMMAIL
Internet e-mail: mhvrcfs@us.ibm.com
World Wide Web: http://www.s390.ibm.com/os390

If you would like a reply, be sure to include your name, address, telephone number, or FAX number.

Make sure to include the following in your comment or note:

- Title and order number of this book
- Page number or topic related to your comment

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Notices

Trademarks

IEEE	Institute of Electrical and Electronics Engineers
ISO	International Organization for Standardization
POSIX	Institute of Electrical and Electronics Engineers

Summary of Changes

Summary of Changes
for GC28-1897-04
OS/390 Release 4

This book contains information previously presented in OS/390 OpenEdition XPG4 Conformance Document, GC23-1897-02, which supports OS/390.

The following summarizes the changes to that information.

New Information

There is no new information for this release.

Changed Information

Minor changes were made throughout the book. Attachment C was changed.

Summary of Changes
for GC28-1897-03
OS/390 Release 3

This book contains information previously presented in OS/390 OpenEdition XPG4 Conformance Document, GC23-1897-02, which supports OS/390.

The following summarizes the changes to that information.

New Information

There is no new information for this release.

Changed Information

Various appendixes contain changes for this release.

Summary of Changes
for GC28-1897-02
OS/390 Release 2

This book contains information previously presented in OS/390 OpenEdition XPG4 Conformance Document, GC23-1897-01, which supports OS/390.

The following summarizes the changes to that information.

New Information

Chapters on the Transport Service (XTI), XPG4 Sockets, and XPG4 Internationalized Terminal Interfaces have been added. In addition, the following attachments were added:

- Attachment E describes the software environment in which the X/Open vsu4 Test Suite was loaded, installed, configured, built, and executed.

- Attachment F describes the hardware environment in which the X/Open vsu4 Test Suite was loaded, installed, configured, built, and executed.
- Attachment G describes the software environment in which the X/Open vst4 master and slave test suites were loaded, installed, configured, built, and executed.

Summary of Changes for GC28-1897-01 OS/390 Release 2

This book contains information previously presented in OS/390 OpenEdition XPG4 Conformance Document, GC23-1897-00, which supports OS/390.

The following summarizes the changes to that information.

Changed Information

This document has been revised to include a new version of the XPG4 Component: XPG4 Commands and Utilities. The new version is called XPG4 Component: XPG4 Command and Utilities V2. Attachments A, B, C, and D were changed.

Summary of Changes for GC28-1897-00 OS/390 Release 1

This book contains information previously presented in *OpenEdition MVS XPG4 Conformance Document*, GC23-3873, which supports MVS/ESA.

The following summarizes the changes to that information.

New Information

Information about the **SRC/common/vtools/y.tab.c** file has been changed.

Changed Information

None

This book includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

XPG4 UNIX Profile

X/Open Conformance Statement

Type: XPG4 Profile

Profile Name: XPG4 UNIX

Completed by:	International Business Machines Corporation
	(name and organization)
on:	February 4, 1998
	(date)

XPG4 UNIX

Product Identification

Product Identification	OS/390
Version/Release No.	Version 2 Release 4 or later

If you do not supply this product yourself, please identify below the supplier you reference:

With:

OS/390 V2R4 or later Security Server
OS/390 V2R4 or later C/C++ Compiler

Environment Specification

a. Binary-compatible Family:

    IBM System/390 processors that support OS/390 Version 1 Release 2 or later.
    See Attachment B.

b. Special instructions for configuring the Product(s) to meet the Conformance Requirements of this profile:

    See Attachment C.
    See Attachment F.

Temporary Waivers

None.

1.1 Components

Completed CSQs for all the following XPG4 components are attached.

- XPG4 Internationalized System Calls and Libraries Extended
- XPG4 Commands and Utilities V2. This includes mandatory conformance to the X/Open UNIX Extension feature group (the **cc** and **c89** command extensions).
- XPG4 C Language
- XPG4 Transport Service (XTI)
- XPG4 Sockets
- XPG4 Internationalized Terminal Interfaces

XPG4 Internationalized System Calls and Libraries Extended Profile

X/Open Conformance Statement
(Revised July 1996)

Type: XPG4 Component

Profile Name: XPG4 Internationalized System Calls and Libraries Extended

Completed by:	International Business Machines Corporation
	(name and organization)
Signed:	
on:	February 4, 1998
	(date)

XPG4 Internationalized System Calls and Libraries Extended

Product Identification

Product Identification	OS/390
Version/Release No.	Version 2 Release 4 or later

With:

Version 2 Release 4 or later Security Server
Version 2 Release 4 or later C/C++ Compiler

Indicator of Compliance

Test report from — VSX4
Test suite release number — 4.3.6
Test report reference number — CTRPOK404
Test Report from VSU4.
Test suite release number — 4.1.1
Test suite reference number —CTRPOK404

Environment Specification

a. Testing Environment:

See Attachment A.
See Attachment C.
See Attachment E.
See Attachment F, OEBRAND1.

b. Binary-compatible Family:

IBM System/390 Processors that support OS/390 Version 1 Release 2 or later.
See Attachment B.

Temporary Waivers

None.

2.1 General Attributes

2.1.1 XPG4 Feature Groups

Question 1: Which of the following Feature Groups are supported by the implementation?

Response:

POSIX.2 C-language Binding	Yes
Shared Memory	Yes
Encryption	Yes
Enhanced Internationalization	Yes
X/Open UNIX Extension	Yes

The POSIX.2 C-language Binding, Shared Memory, Enhanced Internationalization and X/Open UNIX Extension Feature Groups are mandatory for XPG4 Internationalized System Calls and Libraries Extended conformance.

Support for a Feature Group can only be claimed if *all* interfaces in any group behave according to the relevant descriptions in System Interfaces and Headers, Issue 4, Version 2.

The interfaces in the Encryption Feature Group must exist, whether or not the Feature Group is supported, and each interface must either behave according to the description in System Interfaces and Headers, Issue 4, Version 2, or indicate an error, with *errno* set to [ENOSYS].

Rationale:

System Interfaces and Headers, Issue 4, Version 2 states that the system may provide one or more of the Feature Groups listed. XPG4 Components, version 2 states that the POSIX.2 C-language Binding, Shared Memory, Enhanced Internationalization, and X/Open UNIX Extension Feature Groups are mandatory for compliance to the XPG4 Internationalized Systems Calls and Libraries Extended component.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Section 1.2, Conformance and Section 1.3, Feature Groups.

XPG4 Internationalized System Calls and Libraries Extended Component Definition.

2.1.2 POSIX.1 Supported Features

Question 2: Which of the following options, specified in the* <unistd.h> *header file, are available on the system?

Response:

Macro Name	Meaning	Provided
_POSIX_CHOWN_RESTRICTED	**The use of chown() is restricted to a process with appropriate privileges, and to changing the group ID of a file only to the effective group ID of the process or one of its supplementary group IDs.**	**Yes**
_POSIX_NO_TRUNC	**Pathname components longer than {NAME_MAX) generate an error.**	**Yes**
_POSIX_VDISABLE	**Terminal special characters defined in <termios.h> can be disabled using this character value.**	**Yes**
_POSIX_SAVED_IDS	**Each process has a saved set-user-ID and a saved set-group-ID.**	**Yes**
_POSIX_JOB_CONTROL	**Implementation supports job control.**	**Yes**

Rationale:

For a conformant implementation, all of these POSIX features must be provided. In some cases the feature need not be provided for all files or devices supported by the implementation.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 4, Headers, <**unistd.h**>.

2.1.3 Float, Stdio and Limit Values

Question 3: What are the values associated with the following constants specified in the <**float.h**> ***header file?***

Response:

Macro	Meaning	Value
FLT_RADIX	**Radix of the exponent representation.**	**16**
FLT_MANT_DIG	**Number of base-FLT_RADIX digits in the float significand.**	**6**
DBL_MANT_DIG	**Number of base-FLT_RADIX digits in the double significand.**	**14**
LDBL_MANT_DIG	**Number of base-FLT_RADIX digits in the long double significand.**	**28**
FLT_DIG	**Number of decimal digits, *q*, such that any floating point number with *q* digits can be rounded into a float representation and back again without change to the *q* digits.**	**6**
DBL_DIG	**Number of decimal digits, *q* such that any floating point number with *q* digits can be rounded into a double representation and back again without change to the *q* digits.**	**15**
LDBL_DIG	**Number of decimal digits, *q*, such that any floating point number with *q* digits can be rounded into a long double representation and back again without change to the *q* digits.**	**32**
FLT_MIN_EXP	**Minimum negative integer such that FLT_RADIX raised to that power minus 1 is a normalized float.**	**−64**
DBL_MIN_EXP	**Minimum negative integer such that FLT_RADIX raised to that power minus 1 is a normalized double.**	**−64**
LDBL_MIN_EXP	**Minimum negative integer such that FLT_RADIX raised to that power minus 1 is a normalized long double.**	**−64**
FLT_MIN_10_EXP	**Minimum negative integer such that 10 raised to that power is in the range of normalized floats.**	**−78**
DBL_MIN_10_EXP	**Minimum negative integer such that 10 raised to that power is in the range of normalized doubles.**	**−78**
LDBL_MIN_10_EXP	**Minimum negative integer such that 10 raised to that power is in the range of normalized long doubles.**	**−78**

Macro	Meaning	Value
FLT_MAX_EXP	**Maximum integer such that FLT_RADIX raised to that power minus 1 is a representable finite float.**	**63**
DBL_MAX_EXP	**Maximum integer is a representable finite double.**	**63**
LDBL_MAX_EXP	**Maximum integer such that FLT_RADIX raised to that power minus 1 is a representable finite long double.**	**63**
FLT_MAX_10_EXP	**Maximum integer such that 10 raised to that power is in the range of representable finite floats.**	**75**
DBL_MAX_10_EXP	**Maximum integer such that FLT_RADIX raised to that power minus 1 is a representable finite double.**	**75**
LDBL_MAX_10_EXP	**Maximum integer such that 10 raised to that power is in the range of representable finite long doubles.**	**75**
FLT_MAX	**Maximum representable finite float**	**7.2370051E75**
DBL_MAX	**Maximum representable finite double**	**7.2370051E75**
LDBL_MAX	**Maximum representable finite long double**	**7.2370051E75**
FLT_EPSILON	**Difference between 1.0 and the least value greater than 1.0 that is representable as a float.**	**9.536743e-07**
DBL_EPSILON	**Difference between 1.0 and the least value greater than 1.0 that is representable as a double.**	**2.220446e-16**
LDBL_EPSILON	**Difference between 1.0 and the least value greater than 1.0 that is representable as a long double.**	**3.081488e-33**
FLT_MIN	**Minimum normalized positive float**	**5.397605e-79**
DBL_MIN	**Minimum normalized positive double**	**5.397605e-79**
LDBL_MIN	**Minimum normalized positive long double**	**5.397605e-79**

Rationale:

This set of constants provides useful information regarding the underlying architecture of the implementation.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 4, Headers, <**float.h**>

Question 4: What are the values associated with the following constants (optionally specified in the <**limits.h**> ***header file)?***

Macro Name	Meaning	Value
ARG_MAX	**Maximum length of argument to the *exec* functions, including the environment data.**	**1 048 576**
CHILD_MAX	**Maximum number of processes per user ID.**	**32 767**
LINK_MAX	**Maximum number of links to a single file.**	**65 536**
MAX_CANON	**Maximum number of bytes in a terminal canonical input line**	**255**
MAX_INPUT	**Maximum number of bytes for which space will be available in a terminal input queue**	**255**
MAX_MAX	**Maximum number of bytes in a filename (not including the terminating null)**	**255**
OPEN_MAX	**Maximum number of open files that one process can have open at any one time**	**65 535**
PATH_MAX	**Maximum number of bytes in a pathname (including the terminating null)**	**1023**
PIPE_BUF	**Maximum number of bytes that is guaranteed to be atomic when writing to a pipe**	**16 384**
STREAM_MAX	**Number of streams that one process can have open at one time**	**1000**
TZNAME_MAX	**Number of bytes supported for the name of a time zone**	**9**

Rationale:

Each of these limits can vary within bounds set by System Interfaces and Headers, Issue 4, Version 2. The minimum permitted value is specified in Chapter 4, <**limits.h**> of X/Open CAE Specifications, System Interfaces and Headers, Issue 4, Version 2.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 4, Headers, <**limits.h**>

Question 5: What are the values associated with the following constants specified in the <**limits.h**> ***header file?***

Macro Name	Meaning	Minimum	Maximum
BC_BASE_MAX	**Minimum** *ibase* **and** *obase* **values allowed by the bc utility**	**99**	**99**
BC_DIM_MAX	**Maximum number of elements permitted in an array by the bc utility**	**2048**	**2048**
BC_SCALE_MAX	**Maximum scale value allowed by the bc utility**	**99**	**99**
BC_STRING_MAX	**Maximum length of a string constant accepted by the bc utility**	**1000**	**1000**
COLL_WEIGHTS_MAX	**Maximum number of weights that can be assigned to an entry of the LC_COLLATE order keyword in the locale definition file.**	**2**	**2**
EXPR_NEST_MAX	**Maximum number of expressions that can be nested within parentheses by the expr utility.**	**32**	**32**
LINE_MAX	**Maximum length in bytes including the trailing newline of a utility's input line when the utility is described as processing text files.**	**2048**	**2048**
NGROUPS_MAX	**Maximum number of simultaneous supplementary group IDs per process.**	**0**	**300**
RE_DUP_MAX	**Maximum number of repeated occurrences of a regular expression permitted when using interval notation.**	**255**	**255**

Rationale:

Each of these limits can vary within bounds set by System Interfaces and Headers, Issue 4, Version 2. The minimum value that a limit can take on any conforming system is given in the corresponding _POSIX_ or _POSIX2_ value. A specific conforming implementation may provide a higher minimum value than this and the maximum value that it provides can differ from the minimum. Some conforming implementations may provide a potentially infinite value as the maximum, in which case the value is considered to be indeterminate. The minimum value must always be definitive since the _POSIX_ or _POSIX2_ value provides a known lower bound for the range of possible values.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 4, Headers, <**limits.h**>.

Question 6: What are the values associated with the following numerical constants specified in the* >limits.h> *header file?

Response:

Macro Name	**Meaning**	**Value**
CHAR_BIT	**Number of bits in a char**	**8**
CHAR_MAX	**Maximum value of a char**	**255**
INT_MAX	**Maximum value of an int**	**2 147 483 647**
LONG_BIT	**Number of bits in a long int**	**32**
LONG_MAX	**Maximum value of a long int**	**2 147 483 647**
MB_LEN_MAX	**Maximum number of bytes in a character, for any supported locale**	**4**
SCHAR_MAX	**Maximum value of a signed char**	**127**
SHRT_MAX	**Maximum value of a short**	**32 767**
SSIZE_MAX	**Maximum value of an object of type ssize_t**	**2 147 483 647**
UCHAR_MAX	**Maximum value of an unsigned char**	**255**
UINT_MAX	**Maximum value of an unsigned int**	**4 294 967 295**
ULONG_MAX	**Maximum value of an unsigned long int**	**4 294 967 295U**
USHRT_MAX	**Maximum value of an unsigned short int**	**65 535**
WORD_BIT	**Number of bits in a word or int**	**16**

Rationale:

This set of constants provides useful information regarding the underlying architecture of the implementation.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 4, Headers, <**limits.h**>.

Question 7: What are the values associated with the following numerical constants specified in the* <stdio.h> *header file?

Response:

Macro Name	Meaning	Value
FILENAME_MAX	**Maximum size in bytes of the longest filename string that the implementation guarantees can be opened.**	**1024**
FOPEN_MAX	**Number of streams which the implementation guarantees can be open simultaneously.**	**64**
L_ctermid	**Maximum size of character array to hold ctermid() output.**	**1024**
L_tmpnam	**Maximum size of character array to hold tmpnam() output.**	**1024**
TMP_MAX	**Minimum number of unique filenames generated by tmpnam(), which is the maximum number of times an application can call tmpnam() reliably.**	**10 000**

Rationale:

This set of constants provide useful information about the implementation.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 4, Headers, <**stdio.h**>.

2.1.4 Error Conditions

Question 8: Which of the following option errors listed in System Interfaces and Headers, Issue 4, Version 2 are detected in the circumstances specified?

Response:

Function	Error	Detected
access()	EINVAL	Yes
	ETXTBSY	No
acos()	EDOM	Yes
asin()	EDOM	Yes
	ERANGE	Yes

Function	Error	Detected
atan()	EDOM	No
	ERANGE	No
atan2()	EDOM	Yes
	ERANGE	Yes
catclose()	EBADF	Yes
	EINTR	Yes
catgets()	EBADF	Yes
	EINTR	Yes
catopen()	EACCES	Yes
	EMFILE	Yes
	ENAMETOOLONG	Yes
	ENFILE	Yes
	ENOENT	Yes
	ENOMEM	Yes
	ENOTDIR	Yes
ceil()	EDOM	No
cfsetispeed()	EINVAL	No
cfsetospeed()	EINVAL	No
chmod()	EINVAL	No
chown()	EINVAL	No
closedir()	EBADF	Yes
	EINTR	Yes
cos()	EDOM	No
erf()	EDOM	No
	ERANGE	Yes
erfc()	EDOM	No
	ERANGE	Yes
exec	ENOMEM	Yes
	ETXTBSY	No
exp()	EDOM	No
	ERANGE	Yes
fabs()	EDOM	No
	ERANGE	No
fclose()	ENXIO	No
fcntl()	EDEADLK	Yes
fdopen()	EBADF	Yes
	EINVAL	Yes
	EMFILE	No
	ENOMEM	No
fflush()	ENXIO	No
fgetc()	ENOMEM	No
	ENXIO	No
fgetpos()	EBADF	Yes
	ESPIPE	Yes
fgetwc()	ENOMEM	No
	ENXIO	No
	EILSEQ	Yes

Function	Error	Detected
fileno()	EBADF	Yes
floor()	EDOM	No
fmod()	EDOM	Yes
	ERANGE	Yes
fopen()	EINVAL	Yes
	EMFILE	No
	ENOMEN	No
	ETXTBSY	No
fork	ENOMEM	Yes
fpathconf()	EBADF	Yes
	EINVAL	Yes
fprintf()	ENVAL	No
	EILSEQ	No
fputc()	ENOMEM	No
	ENXIO	No
fputwc()	ENOMEM	No
	ENXIO	No
	EILSEQ	Yes
freopen()	EINVAL	Yes
	ENOMEM	No
	ENXIO	Yes
	ETXTBSY	No
frexp()	EDOM	No
fscanf()	EILSEQ	No
	EINVAL	No
	ENOMEM	No
	ENXIO	No
fsetpos()	EBADF	Yes
	ESPIPE	Yes
ftw()	EINVAL	No
getcwd()	EACCES	Yes
	ENOMEM	No
getgrgid()	EIO	No
	EINTR	No
	EMFILE	No
	ENFILE	No
getgrnam()	EIO	No
	EINTR	No
	EMFILE	No
	ENFILE	No
getlogin()	EMFILE	Yes
	ENFILE	Yes
	ENXIO	Yes
getpass()	EINTR	No
	EIO	No
	EMFILE	No
	ENFILE	No
	ENXIO	No

Function	Error	Detected
getpwnam()	EIO	No
	EINTR	No
	EMFILE	No
	ENFILE	No
getpwuid()	EIO	No
	EINTR	No
	EMFILE	No
	ENFILE	No
hcreate()	ENOMEM	Yes
hsearch()	ENOMEM	Yes
hypot()	EDOM	No
	ERANGE	Yes
iconv()	EBADF	Yes
iconv_close()	EBADF	Yes
iconv_open()	EMFILE	No
	ENFILE	No
	ENOMEM	No
	EINVAL	No
isatty()	EBADF	No
	ENOTTY	Yes
j0()	EDOM	No
	ERANGE	Yes
j1()	EDOM	No
	ERANGE	Yes
jn()	EDOM	No
	ERANGE	Yes
ldexp()	EDOM	No
	ERANGE	Yes
lgamma()	EDOM	Yes
	ERANGE	Yes
log()	EDOM	Yes
	ERANGE	Yes
log10()	EDOM	Yes
	ERANGE	Yes
mblen()	EILSEQ	No
mbstowcs()	EILSEQ	No
mbtowc()	EILSEQ	No
modf()	EDOM	No
	ERANGE	No
open()	EINVAL	Yes
	ETXTBSY	No
opendir()	EMFILE	Yes
	ENFILE	Yes
pathconf()	EACCES	Yes
	EINVAL	Yes
	ENAMETOOLONG	Yes
	ENOENT	Yes
	ENOTDIR	Yes

Function	Error	Detected
popen()	EMFILE	No
	EINVAL	Yes
pow()	EDOM	Yes
	ERANGE	Yes
putenv()	ENOMEM	Yes
read()	ENXIO	No
readdir()	EBADF	Yes
rename()	ETXTBSY	No
setvbuf()	EBADF	No
sigaction()	EINVAL	Yes
sigaddset()	EINVAL	Yes
sigdelset()	EINVAL	Yes
sigismember()	EINVAL	Yes
signal()	EINVAL	No
sin()	EDOM	No
	ERANGE	Yes
sinh()	EDOM	No
	ERANGE	Yes
sqrt()	EDOM	Yes
strcoll()	EINVAL	No
strerror()	EINVAL	No
strtod()	EINVAL	No
strtol()	EINVAL	Yes
strtoul()	EINVAL	Yes
strxfrm()	EINVAL	No
system()	ECHILD	No
tan()	EDOM	No
	ERANGE	Yes
tanh()	EDOM	No
	ERANGE	Yes
tcdrain()	EIO	No
tcflow()	EIO	Yes
tcflush()	EIO	No
tcsendbreak()	EIO	No
tcsetattr()	EIO	No
tmpfile()	EMFILE	Yes
	ENOMEM	Yes
ttyname()	EBADF	No
	ENOTTY	No
ungetwc()	EILSEQ	No
unlink()	ETXTBSY	No
wcscoll()	EINVAL	No

Function	Error	Detected
wcstod()	EINVAL	No
wcstol()	EINVAL	Yes
wcstombs()	EILSEQ	No
wcstoul()	EINVAL	No
wcsxfrm()	EINVAL	No
write()	ENXIO	No
y0()	EDOM	Yes
	ERANGE	Yes
y1()	EDOM	Yes
	ERANGE	Yes
yn()	EDOM	Yes
	ERANGE	Yes

Rationale:

Each of the above error conditions is marked as optional in System Interfaces and Headers, Issue 4, Version 2 and an implementation may return this error in the circumstances specified or may not provide the error indication.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Section 2.3, Error Numbers.

2.1.5 Mathematical Interfaces

Question 9: What format of floating-point numbers is supported by this implementation?

Response:

IBM floating point formats correspond most closely to the IEEE single format for binary floating point numbers. IBM floating point format provides more significant digits in the fraction (mantissa) and a wider range on the exponent than the IEEE single format, as described below:

IEEE Single Format Representation: (uses base 2)

```
v = (-1**s)(1xf)(2**e-127)
```

where:

v = binary floating point number
f = fraction (mantissa), stored in 23 bits
s = sign bit
e = biased exponent, stored in 8 bits
    range of e: (0,255)
    range of (e-127): (-127,128)

IBM Floating Point Representative (uses base 16)

```
v = (-1**s)(1xf)(16**e-64) = (-1**s)(1xf)(2**(4e-256))
```

where:

v = floating point number
f = fraction (mantissa)
  IBM short floating point—stored in 24 bits
  IBM long floating point—stored in 56 bits
  IBM extended floating point—stored in 112 bits
s = sign bit
e = biased exponent, stored in 7 bits
  range of e: (0,127)
  range of (4e-256): (-256,252)

Rationale:

Most implementations support IEEE floating point format either in hardware or software. Some implementations support other formats with different exponent and mantissa accuracy. These differences need to be defined.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Section 1.6, Relationship to Formal Standards.

2.1.6 Data Encryption

Question 10: Are the optional data encryption interfaces provided?

Response:

Function	**Provided**
crypt()	Yes
encrypt()	Yes
setkey()	Yes

The full function (two-way) encryption feature will require a special State Department licensing (IVL) to be exported outside the USA.

Rationale:

Normally, an implementation will either provide all three of these routines or will provide none of them at all. If the routines are not provided, then the implementation must provide a dummy interface which always raises an ENOSYS error condition.

It is also possible that the implementation of the **encrypt()** function may be affected by export restrictions, in which case, the restrictions should be documented here.

For example, historical implementations have supplied all three of these routines outside the U.S.A., but due to export restrictions on the decoding algorithm, a dummy version of **encrypt()** is provided that does encoding but no decoding.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Section 1.2, Conformance.

2.2 Process Handling

2.2.1 Process Generation

Question 11: Which file types (regular, directory, FIFO, special, and so on) are considered to be executable?

Response:

Only regular file types may be executed.

Rationale:

The [EACCES] error associated with *exec* functions occurs in circumstances when the implementation does not support execution of files of the type specified. A list of these file types needs to be provided.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 3, System Interfaces, **exec**.

2.3 File Handling

2.3.1 Access Control

Question 12: What file access control mechanisms does the implementation provide?

Response:

Standard access control is provided.

Rationale:

System Interfaces and Headers, Issue 4, Version 2 notes that implementations may provide *additional* or *alternate* file access control mechanisms, or both.

Reference:

X/Open CAE Specification, System Interface Definitions, Issue 4, Version 2, Chapter 2, Glossary, file access permissions.

2.3.2 Files and Directories

Question 13: Are any additional or alternate file access control mechanisms implemented that could cause **fstat()** ***or*** **stat()** ***to fail?***

Response:

No

Rationale:

System Interfaces and Headers, Issue 4, Version 2 notes that there could be an interaction between additional and alternate access controls and the success of **fstat()** and **2stat()**. This would suggest that an implementation can allow access to a file but not allow the process to gain information about the status of the file.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 3, System Interfaces, **fstat()** and **stat()**.

2.3.3 Formatting Interfaces

Question 14: Does the **printf()** ***function produce character string representations for Infinity and NaN to represent the respective values?***

Response:

No

Rationale:

This behavior is often provided on systems with mathematical functions that produce these results.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 3, System Interfaces, **fprintf()**.

2.4 Internationalized System Interfaces

2.4.1 Coded Character Sets

Question 15: What coded character sets are supported by the implementation?

Response:

IBM-037
IBM-273
IBM-274
IBM-275
IBM-277
IBM-278
IBM-280
IBM-281
IBM-282
IBM-284
IBM-285
IBM-290
IBM-297
IBM-500
IBM-871
IBM-875

IBM-930
IBM-939
IBM-1026
IBM-1027
IBM-1047

Rationale:

System Interface Definitions, Issue 4, Version 2 states that conforming implementations support one or more coded character sets, and that each of these includes the portable character set.

Reference:

X/Open CAE Specification, System Interface Definitions, Issue 4, Version 2, Chapter 4, Character Set.

Question 16: What is the implementation's underlying internal codeset?

Response:

EBCDIC (IBM-1047)

Rationale:

It is useful to be aware of the underlying codeset of the implementation.

Reference:

X/Open CAE Specification, System Interface Definitions, Issue 4, Version 2, Chapter 4, Character Set.

2.5 STREAMS

Question 17. What networking services or other character-based I/O device types are implemented using STREAMS?.

Response: NONE

Rationale:

System Interfaces and Headers, Issue 4, Version 2, defines that STREAMS provide a uniform mechanism for implementing networking services and other character-based I/O. However, the specification does not mandate which device or file types should be STREAMS-based. Although applicatons are discouraged from making assumptions in this area, it may be that certain applications are sensitive to whether i interfaces such as **getmsg()** and **putmsg()**, for example, are supported on specific device or file types.

Reference

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Section 2.5, STREAMS.

2.6 Pseudo-Terminals

2.6.1 Master close

Question 18. Does closing the master side of a pseudo-terminal flush all queued input and output?

Response: Yes

Rationale:

The behaviour of a conforming implementation in this area is not mandated in the specification and needs to be defined.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 3, System Interfaces, **close()**.

2.6.2 Slave Close

Question 19. Does closing the slave side of a pseudo-terminal cause a zero-length message to be sent to the master?

Response: Yes

Rationale:

The behaviour of a conforming implementation in this area is not mandated in the specification and needs to be defined.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 3, System Interfaces, **close()**.

2.6.3 Naming Convention

Question 20. What naming conventions are associated with the master side of pseudo-terminal devices?

Response:

```
/dev/ptypNNNN
```

where NNNN is between 0000 and 9999.

Note: 9999=one less than maximum allowable value of MAXPTYS value in the BPXPRMxxx parmlib member. MAXPTYS specifies the maximum number of pseudo TTY sessions that can be active at the same time. The range is 1 to 10 000; the default is 256.

Rationale:

The information is not specified in System and Interfaces, Issue 4, Version 2, and needs to be defined.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 3, System Interfaces, **open()**.

2.7 Polling

Question 21: What types of files can be polled?

Response:

Regular files
Terminals
Pseudo-terminals
Sockets
FIFOs
Pipes

Rationale:

Conformance requires that the **poll()** function supports regular files, terminals, pseudo-terminals, STREAMS, sockets, FIFOs, and pipes. The behavior of **poll()** with regards to othe file types needs to be defined.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 3, System Interfaces, **poll()**.

2.8 Alternate Stacks

Question 22: What allocation routine(s) is provided for creating alternate stack areas?

Response:

calloc()
malloc()
valloc()

Rationale:

Conformance requires that an implementation supports alternate signal stacks. The APPLICATION USAGE section of the **sigalstack()** entry describes one method using **malloc()** to perform this function. However, the specification does not guarantee that malloc'ed space can be used in this way, nor does it define a specific alternate stack allocation routine.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 3, System Interfaces, **sigaltstack()**.

2.9 Signal Codes

Question 23. Which of the following si_code values may be generated?

Response: The **si_code** levels are generated for all of the signals in the following table:

Signal	Code
SIGILL	ILL_ILLOPC ILL_ILLOPN ILL_ILLADDR ILL_ILLTRP ILL_PRVOPC ILL_PRVREG ILL_COPROC ILL_BADSTK
SIGFPE	FPE_INTDIV FPE_INTOVF FPE_FLTDIV FPE_FLTOVF FPE_FLDUND FPE_FLTRES FPE_FLTINV FPE_FLTSUB
SIGSEGV	SEGV_MAPPERR SEGV_ACCERR
SIGBUS	BUS_ADRALN BUS_ADRERR BUS_OBJERR
SIGCHLD	CLD_EXITED CLD_KILLED CLD_DUMPED CLD_TRAPPED CLD_STOPPED CLD_CONTINUED
SIGPOLL	POLL_IN POLL_OUT POLL_MSG POLL_ERR POLL_PRI POLL_HUP

Rationale:

An XPG4 Internationalized System Calls and Libraries Extended conformant system may contain limitations that prevent some of the above values from being generated.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 3, System Interfaces, <**signal.h**>.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 3, System Interfaces, <**signal.h**>.

2.10 Set Process Group ID

Question 24: Does the setpgrp() function create a new session?

Response: No

Rationale:

It is unspecified whether or not a successful call to the setpgrp() function will cause a new session to be created.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 3, System Interfaces, **setgrp()**.

Question 25: Does the implementation provide a signal, when delivered to a process, that generates a core file?

Response: No

Rationale:

Implementation-dependent abnormal termination actions, such as creation of a core file, may occur.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 3, System Interfaces, <**signal.h**>.

XPG4 Commands and Utilities Profile

X/Open Conformance Statement
(Revised September 1995)

Type: XPG4 Component

Component Name: XPG4 Commands and Utilities V2

Completed by:	International Business Machines Corporation
	(name and organization)
on:	February 4, 1998
	(date)

XPG4 Commands and Utilities

Product Identification

Product Identification	OS/390
Version/Release No.	Version 2 Release 4 or later

If you do not supply this component yourself, please identify below the supplier you reference:

With:

OS/390 V2R4 or later Security Server
OS/390 V2R4 or later C/C++ Compiler

Indicator of Compliance

Test Report from VSC4.
Test Suite release number: 1.6
Test report reference number: CTRPOK403

Environment Specification

a. Testing Environment:

   See Attachment C.
   See Attachment D.
   See Attachment F, OEBRAND2

b. Binary-compatible family:

   IBM System/390 Processors that support OS/390 Version 1 Release 3 or later.
   See Attachment B.

c. Portability Environment.

   The environment contains an XPG4 Internationalized System Calls and Libraries Extended branded component.

Temporary Waivers

None.

3.1 XPG4 Feature Groups

Question 1: Which of the following feature groups are supported by the implementation?

Response:

X/Open UNIX Extension — Yes

Conformance to this component does not require extensions to the **cc** and **c89** commands identified by the X/Open UNIX Extension feature group to be supported.

However, support of this extended functionality is required for conformance to the XPG4 UNIX profile.

Rationale:

Support fo the UNIX Extension is not required for conformance to this component.

Reference:

XPG4 Commands and Utilities V2 Component Definition
XPG4 UNIX Profile Definition

3.2 POSIX.2 Supported Features

Question 2: Which of the following options, specified in the <unistd.h> header, are available on the system?

Macro Name	**Meaning**	**Provided**
_POSIX2_C_BIND	Implementation supports the C language binding option.	Yes
_POSIX2_C_DEV	Implementation supports the C language development option.	No
_POSIX2_CHAR_TERM	Implementation supports at least one terminal type.	Yes
_POSIX2_FORT_DEV	Implementation supports the FORTRAN Developmental Utilities Option.	No
_POSIX2_FORT_RUN	Implementation supports FORTRAN Run-time Utilities.	No
_POSIX2_LOCALEDEF	Implementation supports the creation of locales by the **localedef** utility.	Yes
_POSIX2_SW_DEV	Implementation supports Software Develop-ments Utilities Option.	Yes
_POSIX2_UPE	Implementation supports the User Portability Utilities Option.	Yes

Rationale:

For an XPG4 Commands and Utilities V2 conformant implementation, _POSIX2_C_BIND, _POSIX2_CHAR_TERM, _POSIX2_LOCALEDEF, _POSIX2_UPE must be supported. The other constants identify optional functionality that an implementation may or may not choose to support.

Reference:

X/Open CAE Specification, System Interfaces and Headers, Issue 4, Version 2, Chapter 4, Headers, <**unistd.h**>

3.3 Development Utilities

3.3.1 Supported Commands

Question 3: Which of the development utilities are not provided with the implementation?

Response:

None of the development XPG4 utilities are provided.

Rationale:

The development utilities are required to exist on designated DEVELOPMENT systems but may not be present on all XSI-conformant systems. The **dis** utility is defined as optional and need not be present even on systems that support the remainder of the development utilities.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Version 2, Section 1.3.1, Optional, and Section 1.3.2, Development.

3.4 Fortran Option

3.4.1 Fortran Utility

Question 4: Is the FORTRAN **fort77** ***utility provided?***

Response:

No.

Rationale:

The **fort77** utility is the command-level interface to the FORTRAN compiler, which need not be provided.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Section 1.3.3, FORTRAN

3.5 Possibly Unsupportable Utilities and Options

Question 5: Which of the following utilities and utility options are not supported on the implementation?

Utility	Option	Supported
ar	**–s**	Yes
cancel		No
cu		No
lp	**–m**	Yes
lp	**–o**	Yes
lp	**–t**	Yes
lp	**–w**	Yes
lpstat		No
sort	**–z**	Yes
tabs	**+m**	Yes
uucp		Yes
uulog		No
uuname		No
uupick		No
uustat		Yes
uuto		No
uux		Yes

Rationale:

A number of utilities and utility options are marked as possibly unsupportable features, and the functionally associated with these need not be present in a conforming implementation.

Reference: X/Open CAE Specification, Commands and Utilities, Issue 4, Section 1.7, Portability.

3.6 Specific Commands and Utilities

3.6.1 at

Question 6: How does the* at *command interpret a non-null* SHELL *environment variable?

Response:

Uses the shell specified in the **SHELL** environment variable.

Rationale:

The interpretation of the **SHELL** environment variable can cause **at** to invoke different versions of the shell on some implementations.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **at**, ENVIRONMENT VARIABLES, **SHELL**.

3.6.2 awk

Question 7: What is the limit on the number of open streams provided by* awk*?

Response:

3996 of open streams.

Rationale:

The number of open streams that are available to **awk** may differ between implementations, possibly depending on the number of streams that are available to a process ({FOPEN_MAX}).

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **awk**, Input/Output and General Functions, **close()**.

3.6.3 batch

Question 8: How does the* batch *command interpret a non-null* SHELL *environment variable?

Response:

Uses the shell specified in the **SHELL** environment variable.

Rationale:

The interpretation of the **SHELL** environment variable can cause **batch** to invoke different versions of the shell on some implementations.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **batch**, ENVIRONMENT VARIABLES, **SHELL**.

3.6.4 c89

Question 9: Which defined names are automatically provided by the compiler?

Response:

```
–D "errno=(*__errno())"
–D _POSIX_SOURCE=1
–D _POSIX1_SOURCE=2
–D _POSIX_C_SOURCE=2
```

Rationale:

The automatic provision of defined names by the compiler can cause these names to be unavailable in the name space for defined names.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **c89**, OPTIONS, **–D**.

Question 10: When multiple input files are specified, where does* c89 *direct identification messages designating the start of each input file processing?

Response:

Standard error.

Rationale:

These messages, if produced, must be written to one or the other of standard output and standard error, but not to both. The destination of these messages is useful in determining redirections that are necessary to identify the input files from which warning messages are generated.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **c89**, STDOUT and STDERR.

Question 11: What are the limits associated with external symbols imposed by* c89*?

Response:

Description	Minimum Maximum	Implementation Maximum
Number of significant bytes	31	255
Number of source or object files	511	65535
Total number of external symbols	4095	65535

Rationale:

These limits vary between implementations and cannot be reset by the user. The XCU definition gives the minimum maximum value for each of the values. Some applications may require larger limits than these minimum maxima.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **c89**, EXTENDED DESCRIPTION, External Symbols.

3.6.5 cancel

Question 12: Is the submitter of an* lp *job notified when the job is canceled by someone else?

Response:

Yes

Rationale:

It is useful for the submitter of a job to be notified of its cancellation, rather than having to check the line printer queue to obtain this information.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **cancel**, ENVIRONMENT VARIABLES, **LANG**.

3.6.6 cp

Question 13: What is the effect of alternate access control mechanisms on file copies?

Response:

No alternate access control mechanisms are implemented.

Rationale:

Because of the additional restrictions on creating files and reading data from files, **cp** utility may not behave as described when alternate access control mechanisms are in use.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **cp**, DESCRIPTION (final paragraph).

3.6.7 date

Question 14: Does* date *permit the setting of the date and time?

Response:

No

Rationale:

Some systems, particularly those that are hosted as part of a total system environment, do not allow the **date** command to set the date. On such systems, the setting of the date can only be accomplished from the host environment.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **date**, OPERANDS, mmddhhmm[yy].

3.6.8 ex

Question 17: What restrictions are imposed on the set of commands within the* rhs *of the* map *command?

Response:

Function Keys are the only things that cannot appear on the rhs of a mapping.

Rationale:

Implementations may impose restrictions on the commands that can be used by macros in visual mode.

Reference:

X/Open CAE Specifications, Commands and Utilities, Issue 4, Chapter 3, Utilities, **ex**, EXTENDED DESCRIPTION, Command Descriptions in ex, Map.

3.6.9 fc

Question 16: Is the history list mechanism disabled for users with appropriate privileges who do not set* HISTFILE*?

Response:

No

Rationale:

XPG4 states that an implementation may, in certain circumstances, disable the history list mechanism for users with appropriate privileges who do not set **HISTFILE**. This could have some security implications.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **fc**, ENVIRONMENT VARIABLES, **HISTFILE**.

3.6.10 fort77

Question 17: When multiple input files are specified, where does* fort77 *direct identification messages designating the start of each input file processing?

Response:

Not applicable.

Rationale:

These messages, if produced, must be written to either standard output or standard error, but not to both. The destination of these messages is useful in determining redirections that are necessary to identify the input files from which warning messages are generated.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **fort77**, STDERR.

Question 18: What are the limits associated with external symbols imposed by* fort77*?

Response:

Description	**Minimum Maximum**	**Implementation Maximum**
Number of significant bytes	31	Not applicable
Number of source or object files	511	Not applicable
Total number of external symbols	4095	Not applicable

Rationale:

These limits vary between implementations and cannot be reset by the user. The specification gives the minimum maximum value for each of the values. Some applications may require larger limits than these minimum maxima.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **fort77**, EXTENDED DESCRIPTION, External Symbols.

3.6.11 lex

Question 19: Where are error messages sent when the* lex –t *option is not specified?

Response:

Standard error.

Rationale:

These messages can be directed to either standard output or standard error according to XPG4, though the messages are not allowed to be directed to both. An application may wish to redirect these messages to a file.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **lex**, STDOUT.

3.6.12 ln

Question 20: Can* ln *create links to a directory?

Response:

No

Rationale:

Implementations may disallow the creation of hard links to a directory, even though the executing process has the appropriate privileges.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **ln**, OPERANDS, *source_file*.

3.6.13.localedef

Question 21: What is the default character mapping used when the localedef –f option is not specified?

Response:

POSIX portable character set as per IBM-1047

Rationale:

The specification does not define a specific character mapping as the default for conforming systems. This character mapping provides encoding information for the members of the portable character set.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **localedef**, OPTIONS, **–f**.

3.6.14 lp

Question 22: What* lp *option or operator command is used to suppress the printing of a banner page?

Response:

None

Rationale:

The user may require that banner pages be suppressed in cases where preprinted forms are used and the stationery is of a non-standard length.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **lp**, DESCRIPTION.

3.6.15 ls

Question 25: How many bytes are in a block as reported by* ls*?

Response:

512 bytes in a block.

Rationale:

The block size used by **ls** to report the number of blocks occupied by a file varies from system to system. Often this depends on the underlying file system architecture.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **ls**, OPTIONS, **–s**.

3.6.16 make

Question 26: What additional variables does* make *add to its environment?

Response:

DIRSEPSTR
EPILOG
GROUPFLAGS
GROUPSHELL
GROUPSUFFIX
IGNORE
INCDEPTH
MAKECMD
MAKEDIR
MAKEFLAGS
MAKESTARTUP
MFLAGS
NULL
OS
OSRELEASE
OSVERSION
PWD
PRECIOUS
PROLOG
SETDIR
SHELL
SHELLMETAS
SHELLFLAGS
SILENT
SWITCHAR

Rationale:

The implementation of **make** may set certain environment variables on invocation of **make**. These variables may not be set by the user, thus reducing the name space for environment variables.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **make**, EXTENDED DESCRIPTION, Makefile Execution.

Question 25: Does the default* MAKEFLAGS *environment variable contain additional implementation-dependent options?

Response:

–E, **–e**, **–V**, **–v**, **–x**

Rationale:

The implementation of **make** may set certain default **MAKEFLAGS** options on invocation of **make**. These variables are in addition to those set by the user on the command line and could affect the processing of **make**.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **make**, EXTENDED DESCRIPTION, Makefile Execution.

3.6.17 newgrp

Question 26: Does **newgrp** ***allow users who are not listed as a member of a group which has no password to change to that group?***

Response:

No

Rationale:

On some implementations, a user who is not listed as a member of a group can change to that group if there is no password associated with the group.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **newgrp**, DESCRIPTION.

Question 27: Are there any other implementation-specific authorization restrictions that affect **newgrp*****?***

Response:

No

Rationale:

Some implementations may impose accounting or other restrictions that could cause **newgrp** to deny activity to a group member. For example, a resource quota system could be implemented on a group basis that would limit the ability to join a group until the resources were available to the group.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **newgrp**, DESCRIPTION.

3.6.18 nice and renice

Question 28: What are the limits and default values used by **nice** ***and*** **renice*****?***

Response:

Maximum **nice** value is 39.
Minimum **nice** value is 0.
Default **nice** increment is 10.

Rationale:

Each value differs between implementations and the range of values gives the user some control over the relative priority of processes.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **nice**, DESCRIPTION.

3.6.19 pax

Question 29: What is the default archive format used by* pax*?

Response:

Extended **tar**.

Rationale:

The implementation has the choice as to which format it shall use as the default when it is creating files. When it is reading an archive created in either extended **tar** or extended **cpio** format (or any other format that it understands), the **pax** utility will read the archive in the format as written.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **pax**, DESCRIPTION.

Question 30: How does* pax *handle reading and writing of archives that span multiple files?

Response:

The **pax** utility supports multiple volumes and prompts for the user to change them.

Rationale:

In many cases **pax** will take actions, such as prompting the user for the device name to use for the next archive file, when the current archive file is full. There may be extensions to the syntax of **pax** that allow the user to specify the address to use to access subsequent files.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **pax**, DESCRIPTION.

Question 31: How does* pax *handle invalid filenames when it is extracting files from an archive?

Response:

If the **pax** utility cannot extract a file, it issues a message and continues to the next file. The exit status will be `1` and the message issued will be:

```
Failure on extraction
```

Rationale:

An implementation can either extract the data associated with these files into files named in an implementation-defined manner or issue an error indicating that the file is being ignored. If **pax** extracts the file, it is necessary for the user either to be informed of the file that is used or to know the algorithm that **pax** uses in generating these filenames.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **pax**, EXTENDED DESCRIPTION, The cpio Filename.

3.6.20 printf

Question 32: Does* printf *support the* e, E, f, g*, and* G *floating point conversion specifications?

Response:

Yes

Rationale:

The support of these conversions is not required on an XCU conforming system.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **printf**, EXAMPLES.

3.6.21 sh

Question 33: Is the environment variable* IFS *ignored when the shell is invoked?

Response:

Yes

Rationale:

The specification allows that the **sh** command ignore the setting of the **IFS** environment variable on invocation. The setting of this variable has been used to breach security on systems which use the shell to interpret a call to the **system()** and **execvp()** interfaces.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **sh**, ENVIRONMENT VARIABLES, **IFS**.

3.6.22 touch

Question 34: What is the latest date after the Epoch that can be used by* touch*?

Response:

```
00:00:00 EST 01/01/2038
```

Rationale:

Because of the limitations on the storage of times in the **stat** structure associated with a file, there is a limitation on the valid dates that can be specified to **touch**. This is directly related to the value that can be stored in the integral type **time_t**.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **touch**, OPTIONS, **–t**.

3.6.23 yacc

Question 37: What are the limits of* yacc*'s internal tables?

Response:

Description	**Minimum Maximum**	**Implementation Maximum**
Number of tokens	126	Unlimited
Number of nonterminals	200	Unlimited
Number of rules	300	Unlimited
Number of states	600	Unlimited
Length of rules	5200	Unlimited
Number of actions	4000	Unlimited

Rationale:

These internal table sizes vary between implementations and cannot be reset by the user. The XCU definition gives the minimum maximum value for each of the table values.

Reference:

X/Open CAE Specification, Commands and Utilities, Issue 4, Chapter 3, Utilities, **yacc**, EXTENDED DESCRIPTION, Limits.

XPG4 C Language Profile

X/Open Conformance Statement Questionnaire

Type: XPG4 Component

Component Name: XPG4 C Language

Completed by:	International Business Machines Corporation
	(name and organization)
on:	February 4, 1998
	(date)

C Language

ISO C Language

Product Identification

Product Identification	OS/390
| Version/Release No.	Version 2 Release 4 or later

If you do not supply this component yourself, please identify below the supplier you reference:

With:

| OS/390 V2R4 or later Security Server
| OS/390 V2R4 or later C/C++ Compiler

Indicator of Compliance

| Test Report from Perennial ACVS 4.4
| Test suite release number — Version 4.3.6
| Test report reference number — CTRPOK401

Environment Specification

a. Testing Environment:

   See Attachment C.
   See Attachment F, OEBRAND1.

b. Binary-compatible Family:

   IBM System/390 Processors that support OS/390 Version 1 Release 3 or later.
   See Attachment B.

c. Portability Environment:

   The environment contains an XPG4 Internationalized System Calls and Libraries Extended branded component.

Temporary Waivers

None.

4.1 Scope of the Implementation

Question 1: What is the limit on the number of nesting levels of compound statements, iteration control structures and selection control structures?

Response:

Unlimited levels.

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 15 nesting levels of compound statements, iteration control structures and selection control structures, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — C (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 2: What is the limit on the number of nesting levels of conditional inclusions?

Response:

Unlimited levels.

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 8 nesting levels of conditional inclusions, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — C (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 3: What is the limit on the number of pointer, array and function declarators (in any combination) modifying an arithmetic, a structure, a union or an incomplete type in a declaration?

Response:

Unlimited declarers

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 12 pointer, array and function declarators (in any combination) modifying an arithmetic, a structure, a union or an incomplete type in a declaration, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — C (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 4: What is the limit on the number of nesting levels of parenthesized declarators within a full declarator?

Response:

Unlimited levels.

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 31 nesting levels of parenthesized declarators within a full declarator, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages —C (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 5: What is the limit on the number of nesting levels of parenthesized expressions within a full expression?

Response:

Unlimited levels.

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 32 nesting levels of parenthesized expressions within a full expression, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages —C (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 6: What is the number of significant initial characters in an internal identifier or macro name?

Response:

255 characters

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 31 significant initial characters in an internal identifier or macro name, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 7: What is the number of significant initial characters in an external identifier?

Response:

255 (with **LOGNAME**) characters

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 6 significant initial characters in an external identifier, implementations should avoid imposing fixed translation limits whenever possible.

XPG4 XSI-conformant systems support the significance of external identifiers up to a length of at least 31 bytes.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 8: What is the limit on the number of external identifiers in one translation unit?

Response:

Unlimited identifiers

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 511 external identifiers in one translation unit, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 9: What is the limit on the number of identifiers with block scope declared within one block?

Response:

Unlimited identifiers

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 127 identifiers with block scope declared within one block, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 10: What is the limit on the number of macro identifiers simultaneously defined in one translation unit?

Response:

Unlimited identifiers

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 1024 macro identifiers simultaneously defined in one translation unit, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 11: What is the limit on the number of parameters in one function definition?

Response:

Unlimited parameters

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 31 parameters in one function definition, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 12: What is the limit on the number of arguments in one function call?

Response:

Unlimited arguments

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 31 arguments in one function call, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 13: What is the limit on the number of parameters in one macro definition?

Response:

Unlimited parameters.

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 31 parameters in one macro definition, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 14: What is the limit on the number of arguments in one macro invocation?

Response:

Unlimited arguments

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 31 arguments in one macro invocation, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 15: What is the limit on the number of characters in a logical source line?

Response:

32 760 characters

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 509 characters in a logical source line, implementations should avoid imposing fixed translation limits whenever possible. Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 16: What is the limit on the number of characters in a character string literal or wide string literal (after concatenation)?

Response:

4096 characters

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 509 characters in a character string literal or wide string literal (after concatenation), implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 17: What is the limit on the number of bytes in an object (in a hosted environment only)?

Response:

LONG_MAX 2 147 483 647 bytes

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 32 767 bytes in an object (in a hosted environment only), implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 18: What is the limit on the number of nesting levels for* #include*ed files?

Response:

Unlimited levels

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of eight nesting levels for **include**ed files, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 19: What is the limit on the number of* case *labels for a* switch *statement (excluding those for any nested* switch *statement)?

Response:

INT_MAX 2 147 483 647 labels

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 257 **case** labels for a **switch** statement (excluding those for any nested **switch** statement), implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 20: What is the limit on the number of members in a single structure or union?

Response:

Unlimited members

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 127 members in a single structure or union, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 21: What is the limit on the number of enumeration constants in a single enumeration?

Response:

Unlimited contents

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 127 enumeration constants in a single enumeration, iteration control structures, and selection control structures, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

Question 22: What is the limit on the number of levels of nested structure or union definitions in a single struct-declaration-list?

Response:

Unlimited levels

Rationale:

The ISO C specification states that, while the implementation shall be able to translate and execute at least one program that contains at least one instance of 15 levels of nested structure or union definitions in a single struct-declaration-list, implementations should avoid imposing fixed translation limits whenever possible.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 5.2.4.1, Translation Limits.

4.2 Technical Requirements

Question 23: Are distinctions of case ignored in external identifiers?

Response:

Yes.

Rationale:

The ISO C specification states that the implementation may ignore distinctions of case in such names.

Attention: Restriction of the significance of an external name to only one case is an obsolescent feature that is a concession to existing implementations.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 6.1.2, Identifiers.

Question 24: What conversion rules are applied when converting an integral type to a floating type which cannot represent the result exactly?

Response:

Convert to nearest lower value.

Rationale:

The ISO C specification states that when a value of integral type is converted to a floating type, if the value being converted is in the range of values which can be represented but cannot be represented exactly, the result is either the nearest higher value or nearest lower value, chosen in an implementation-defined manner.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 6.2.1, Arithmetic Operands.

Question 25: What conversion rules are applied when converting a* double *to a* float *or a* long double *to a* long float *which cannot represent the result exactly?

Response:

Convert to nearest lower value.

Rationale:

The ISO C specification states that when converting a **double** to a **float** or a **long double** to a **long float**, if the value being converted is in the range of values which can be represented but cannot be represented exactly, the result is either the nearest higher value or nearest lower value, chosen in an implementation-defined manner.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 6.2.1, Arithmetic Operands.

Question 26: What truncation rules are applied when using the division operator and either of the operands is negative?

Response:

Truncation toward zero.

Rationale:

The ISO C specification states that such truncations are machine-dependent.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 6.3.5, Multiplicative Operators.

Question 27: What sign is given to the result when using the remainder operator and either of the operands is negative?

Response:

Remainder has the same sign as that of the dividend.

Rationale:

The ISO C specification states that the sign of the result is machine-dependent.

Reference:

ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 6.3.5, Multiplicative Operators.

Question 28: When mapping sequences of characters to external source file names, does the implementation ignore distinctions of alphabetic case and restrict the mapping to 6 significant characters?

Response:

No

Rationale:

The ISO C specification states that the implementation may ignore distinctions of alphabetic case and restrict the mapping to 6 significant characters.

Reference: ISO/IEC 9899:1990, Programming Languages — (technically identical to ANS X3.159-1989), Section 6.8.2, Source File Inclusion.

X/Open C Language

Product Identification

Product Identification	OS/390
Version/Release No.	Version 1 Release 3 or later

With:

OS/390 V1R2 or later Security Server
OS/390 V1R2 or later C/C++ Compiler

Indicator of Compliance

Test report from VSX4.
Test suite release number—3.5C
Test report reference number—CTRPOK005

Environment Specification

a. Testing Environment:

   See Attachment A.
   See Attachment C.
   See Attachment F, OEBRAND1.

b. Binary-compatible Family:

   See Attachment B

c. The environment contains an XPG4 branded Internationalized Systems Calls and Libraries component.

Temporary Waivers

None.

4.1 Scope of the Implementation

Question 1: What limits does the implementation impose on the significant part of an identifier?

Response:

External identifiers—255 with the compile-time option LONGNAME specified, 8 characters
Nonexternal identifier—255 characters

Rationale:

The XPG states that, while there is no limit to the length of an identifier, only a certain number of characters are significant. The XPG points out that there must be at least eight characters for a nonexternal name, but may be less for external names. On XPG4 XSI-conformant systems this has been extended to support the significance of external identifiers up to a length of at least 31 bytes.

Reference:

X/Open Portability Guide, Issue 3, Volume 4, Programming Languages, Section 2.1, Lexical Conventions.

4.2 Technical Requirements

Question 2: What truncation rules are applied when a floating value is converted to an integral value?

Response:

The fractional part is discarded. The integral part is converted to unsigned long, if converting to unsigned integer type, or to signed long if converting to signed integer type. Then the integral conversions follow the ISO C rules defined in ISO C 6.2.1.2. When converting from integer to a shorter signed integer, or an unsigned integer to a corresponding signed integer, if the value cannot be represented, the bit pattern of the right most bytes of the source value that fit into the target type are unchanged, and the leftmost bytes are truncated.

Rationale:

The XPG states that the conversion of floating values to integral values are machine-dependent. In particular, the XPG points out the differences related to the truncation of negative numbers.

Reference:

X/Open Portability Guide, Issue 3, Volume 4, Programming Languages, Section 2.5, Conversions.

Question 3: What truncation rules are applied when using the division operator and either of the operands is negative?

Response:

Truncation toward zero.

Rationale:

The XPG states that such truncations are machine-dependent.

Reference:

X/Open Portability Guide, Issue 3, Volume 4, Programming Languages, Section 2.6, Expressions.

Transport Service (XTI) Profile

X/Open Conformance Statement Questionnaire
(Revised April 1996)

Type: XPG4 Component

Component Name: XPG4 Transport Service (XTI)

Completed by:	International Business Machines Corporation
	(name and organization)
on:	February 4, 1998
	(date)

XPG4 Transport Service

Product Identification

Product Identification	OS/390
Version/Release No.	Version 2 Release 4 or later

If you do not supply this component yourself, please identify below the supplier you reference:

Indicator of Compliance

Test Report from VST4.
Test suite release number — 1.4
Test report reference number — CTRPOK405

Environment Specification

a. Testing Environment:

   See Attachment F.
   OEBRAND1 was master; OEBRAND2 was slave.
   See Attachment G.
   See Attachment C.

b. Binary-compatible Family:

   IBM System/390 Processors that support OS/390 Version 1 Release 3 or later.
   See Attachment B.

c. Portability Environment:

   The environment contains an XPG4 Internationalized System Calls and Libraries Extended Branded component.

Temporary Waivers

None.

5.1 Optional Features

Question 1: Are the UX extensions in <xti.h> supported?

Response:

Yes

Rationale:

Networking Services, Issue 4 requires that prototypes for the XTI functions are defined in <**xti.h**> if _XOPEN_SOURCE_EXTENDED is defined. This is mandatory for the XPG4 UNIX Profile but is optional otherwise.

Reference:

X/Open CAE Specification, Networking Services, Issue 4, Appendix F, Headers and Definitions for XTI.

5.2 Scope of the Implementation

5.2.1 Transport Providers Supported

Question 2: Which service types and transport providers are supported by the product and what transport provider identifiers are used to access them?

Response:

Service Type	Supported	Transport Provider Identifier(s)
T_COTS	Yes	/dev/tcp
T_COTS_ORD	No	—
T_CLTS	No	—

Description of transport providers: Connection-oriented Transport Service, TCP/IP

Rationale:

The X/Open Transport Interface (XTI) requirements allow three service types (T_COTS, T_COTS_ORD and T_CLTS) which are not all mandatory and these services may be supplied by a number of transport providers. An XTI user process must have knowledge of the service type and the transport provider identifiers supported bya product if it is to make use of an XTI implementation. Transport provider identifiers are supplied in the name parameter of **t_open()** and the service type is supplied in the info->servtype parameter of **t_open()**.

Reference:

X/Open CAE Specification, Networking Services, Issue 4, Chapter 7, XTI Library Functions and Parameters, **t_open()**.

5.3 Technical Requirements

5.3.1 Zero-length Data

Question 3: For which transport providers does the product support the sending of zero-length normal or expedited data transport service data units?

Response:

TCP

Rationale:

Some transport providers forbit the sending of zero-length transport service data units.

Reference:

X/Open CAE Specification, Netowrking Services, Issue 4, Chapter 7, XTI Library Functions and Parameters.

5.4 ISO Connection Mode Transport Provider

Questions 4 and 5 are only relevant for those products which support the ISO Connection-oriented Transport Service as indicated in the response to Question 1.

5.4.1 Quality of Service Options

Question 4: Does the product support negotiations of the following Quality of Service options?

Response:

QoS Parameter	XTI Name	Negotiation Supported
Throughput	TCO_THROUGHPUT	Not Applicable
Transit Delay	TCO_TRANSDEL	Not Applicable
Residual Error Rate	TCO_RESERRORRATE	Not Applicable
Transfer Failure Probability	TCO_TRANSFFAILPROB	Not Applicable
Connection Establishment Failure Probability	TCO_ESTFAILPROB	Not Applicable
Connection Release Failure Probability	TCO_RELFAILPROB	Not Applicable
Connection Establishment Delay	TCO_RELFAILPROB	Not Applicable
Connection Release Delay	TCO_RELDELAY	Not Applicable
Connection resilience Protection	TCO_PROTECTION	Not Applicable
Priority	TCO_PRIORITY	Not Applicable
Expedited Data	TCO_EXPD	Not Applicable

Rationale:

An XTI-compliant transport provider may support none, all, or a subset of the options available for use with the ISO Transport service.

Reference:

ISO 8072:1986, Information Processing Systems — Open Systems Interconnection — Transport Service Definition

X/Open CAE Specification, Networking Services, Issue 4, Section A.2.1.1, Options for Quality of Service and Expedited Data.

5.4.2 Management Options

Question 5: Which of the following XTI management options are supported by the product?

Response:

Option	XTI Name	Supported
Maximum Length of TPDU	TCO_LTPDU	Not applicable
Acknowledge time	TCO_ACKTIME	Not applicable
Reassignment time	TCO_REASTIME	Not applicable

Option	XTI Name	Supported
Preferred Class	TCO_PREFCLASS	Not applicable
1st alternate class	TCO_ALTCLASS1	Not applicable
2nd Alternate Class	TCO_ALTCLASS2	Not applicable
3rd Alternate Class	TCO_ALTCLASS3	Not applicable
4th Alternate Class	TCO_ALTCLASS4	Not applicable
Extended Formats	TCO_EXTFORM	Not applicable
Flow Control	TCO_F:PWCTRL	Not applicable
Checksum Use	TCO_CHECKSUM	Not applicable
Network Expedited Data	TCO_NETEXP	Not applicable
Network Receipt Confirmation	TCO_NETRECPTCF	Not applicable

Rationale:

XTI offers additional management options to those defined as protocol parameters for Quality of Service use in ISO 8072:1986. Not all of these need be supported.

Reference:

X/Open CAE Specification, Networking Services, Issue 4, Section A.2.1.2, Management Options.

5.5 ISO Connectionless Mode Transport Service

Questions 6 and 7 are only relevant to those XTI implementations that support an ISO Connectionless Mode transport provider.

5.5.1 Quality of Service Options

Question 6: Which of the following Quality of Service parameters are supported by the XTI product when an ISO Connectionless Mode transport service is used as the service provider?

Response:

QoS Parameter	XTI Name	Negotiation Supported
Transit Delay	TCL_TRANSDEL	Not Applicable
Residual Error Rate	TCL_RESERRPRRATE	Not Applicable
Protection	TCL_Protection	Not Applicable
Priority	TCL_Priority	Not Applicable

Rationale:

Not all Quality of Service parameters need be supplied by all implementations.

Reference:

X/Open CAE Specification, Networking Services, Issue 4, Section A.2.1.2, Options for Quality of Service.

5.5.2 Management Options

Question 7: Does the product support the use of TCL_CHECKSUM management option to allow user control over whether a checksum is computed for PDUs issued by an ISO Connectionless Mode transport provider?

Response:

Not Applicable.

Rationale:

This management option is not defined in ISO 8072/Add.1:1986 although it does appear as a protocol parameter in ISO 8602. It is offered as an additional option by XTI but need not be supported by all implementations.

Reference:

X/Open CAE Specification, Networking Services, Issue 4, Section A.2.2.2, Management Options.

5.6 TCP Transport Providers

Question 8 need only be answered if support for a TCP transport is indicated in the response to Question 1.

Question 8: Which of the following options are supported for use with TCP transport providers?

Response:

Options	**XTI Name**	**Negotiation Supported**
Check Connections Alive	TCP_KEEPALIVE	Yes
Get Maximum Segment Size	TCP_MAXSEG	No
Don't Delay to Coalesce	TCP_NODELAY	No

Rationale:

A transport provider that is compliant to the XTI specification may support none, all, or any subset of the TCP-level options.

Reference:

X/Open CAE Specification, Networking Services, Issue 4, Section B.2.1, TCP-level Options.

5.7 UDP Transport Provider

Question 9 need only be answered for those implementations that indicated support for a UDP service provider in answer to Question 1.

Question 9: Does the product support user control of the computation of UDP checksums by means of the UDP_CHECKSUM option?

Response:

Not Applicable.

Rationale:

A transport provider compliant to the XTI specification implements none, all, or any of the subset of options defined for TCP, UDP, and IP.

Reference:

X/Open CAE Specification, Networking Services, Issue 4, Section B.2.2, UDP-level Options.

5.8 Internet Protocol Support

Question 10 need only be answered for those implementations that indicated support of the Internet Protocol in their transport providers.

Question 10: Which of the following options are supported at the IP-level by the transport provider(s) associated with the XTI product?

Response:

Options	XTI Name	Supported
Permit Broadcast	IP_BROADCAST	No
Bypass Routing	IP_DONTROUTE	No
IP Per-Packet Options	IP_OPTIONS	No
Local Address Reuse	IP_REUSEADDR	Yes
IP Type of Service	IP_TOS	No
Time To Live	IP_TTL	No

Rationale: Compliant XTI implementations may implement all, none, or any subset of the defined options for control over the IP level of transport providers

Reference:

X/Open CAE Specification, Networking Services, Issue 4, Section B.2.3, IP-level Options.

5.9 Communications Interface

5.9.1 Required Protocol Stack Support

Question 11: Which protocol stacks does the product support?

Response:

Protocol stack	Supported?
ISO ISP 10608-2(TA-51): TP4 and CLNS over LLCI and 8802-3 LAN.	No
ISO ISP 10608-5 (TA 1111): TP4 and CLNS over PSTN SVC.	No
ISO ISP 10609-7 (TD 1111): TPO and CONS over PSTN SVC.	No
TPC/IP	Yes
UDP/IP	No

Rationale:

Products conforming to the XPG4 Transport Service (XTI) component definition must be available in configurations that support at least one of the above transport profiles.

Reference:

XPG4 Transport Service (XTI) Component Definition

X/Open Guide, Guide to the INternet Protocol Suite.

5.9.2 Optional Support for Protocols or Protocol Profiles

Question 12: What other protocols or protocol profiles does your product support?

Response:

None.

Rationale:

The Conformance Statement provides an opportunity to declare support for other protocols or protocol profiles.

Reference:

XPG4 Transport Service (XTI) Component Definition

5.10 Recommended XTI Header Values Implemented

Question 13: Does your implementation use all the recommended values identified in the XTI specification as “recommended only, not mandatory for conformance?”

Response:

Yes.

If “No,” enter the values in the table below:

Symbolic Constant	**Value**
General definitions for option management: T_ALLOP	
XTI-level: XTI_GENERIC	
XTI-level options: XTI-DEBUG XTI-LINGER XTI-RCVBUF XTI-RCVLOWAT XTI-SNDBUF XTI-SNDLOWAT	

Symbolic Constant	**Value**
Protocol levels: ISO_TP	
Options for QoS and expedited data (ISO 8072:1986): TCO_THROUGHPUT TCO_TRANSDEL TCO_RESERRORRATE TCO_TRANSFAILPROB TCO_ESTFAILPROB TCO_RELFAILPROB TCO_ESTDELAY TCO_RELDELAY TCO_CONNRESIL TCO_PROTECTION TCO_PRIORITY TCO_EXPD TCL_TRANSDEL TCL_RESERRORATE TCL_PROTECTION TCL_PRIORITY	—
Management options: TCO_LTPDU TCO_ACKTIME TCO_REASTIME TCO_EXTFORM TCO_FLOWCTRL TCO_CHECKSUM TCO_NETEXP TCO_NETRECPTCF TCO_PREFCLASS TCO_ALTCLASS1 TCO_ALTCLASS2 TCO_ALTCLASS3 TCO_ALTCLASS4 TCL_CHECKSUM	—
TCP-level: INET_TCP	—
TCP-level options: TCP_NODELAY TCP_MAXSEG TCP_KEEPALIVE	

Symbolic Constant	Value
UDP-level: INET_UDP	
UDP-level options:	
UDP_CHECKSUM	
IP-level	.
INET_IP	
IP-level options:	
IP_OPSIONS	
IP_TOS	
IP_TTL	
IP_REUSEADDR	
IP_DONTROUTE	
IP_BROADCAST	

Rationale:

An XTI transport provider does not have to implement those recommended values in order to be compliant. However, it is valuable to know this so that users can easily check whether an XTI application that relies on these values will be portable to this XTI implementation: if the answer is “Yes,” then such an XTI application will be portable to this implementation; if “No,” then it will not.

Reference:

X/Open Corrigendum U008 to X/Open CAE Specification, Networking Services, Issue 4. This Corrigendum revises Appendix F, XTI Headers and Definitions, to define certain values assigned to constants as “recommended, not mandatory.”

XPG4 Sockets Profile

X/Open Conformance Statement

Type: XPG4 Component

Component Name: XPG4 Sockets

Completed by:	International Business Machines Corporation
	(name and organization)
on:	February 4, 1998
	(date)

XPG4 Sockets

Product Identification

Product Identification	OS/390
Version/Release No.	Version 2 Release 4 or later

If you do not supply this component yourself, please identify below the supplier you reference:

With:

OS/390 V2R4 or later Security Server
OS/390 V2R4 or later C/C++ Compiler

Indicator of Compliance

Test Report from VSU4.
Test suite release number — 4.1.1
Test report reference number — CTRPOK402

Environment Specification

a. Testing Environment:

See Attachment C.
See Attachment E.
See Attachment F, OEBRAND1.

b. Binary-compatible Family:

IBM System/390 Processors that support OS/390 Version 1 Release 3 or later.
See Attachment B.

c. Portability Environment:

The environment contains an XPG4 Internationalized System Calls and Libraries Extended Branded component.

Temporary Waivers

None.

6.1 Supported Features

6.1.1 Socket Domains

Question 1: Which socket domains are supported by the implementation?

Response:

Domain	Type	Protocol
AF_INET	SOCK_STREAM	TCP
	SOCK_DGRAM	TCP
AF_UNIX	SOCK_STREAM	0
	SOCK_DGRAM	0

Rationale:

The X/Open CAE Specification, Networking Services, Issue 4, defines that the *domains*, *socket types*, and *protocols* supported by a conforming system are implementation-dependent. XPG4 Component Definitions, Issue 2, states that products conforming to the XPG4 Sockets component definition shall be available in configurations that support the following socket domains:

- AF_INET, with at least SOCK_STREAM and SOCK_DGRAM socket types
- AF_UNIX, with at least the SOCK_STREAM socket type.

Reference:

X/Open CAE Specification, Networking Services, Issue 4, Chapter 8, Sockets Interfaces, *socket()*.

XPG4 Sockets Component Definition

6.2 Limits

6.2.1 Listen Queues

Question 2: What is the limit the implementation places on the length of a socket's listen queue?

Response:

- SOMAXCONN, as defined in **sys/socket.h**.
- For AF_UNIX sockets, this value is variable and can be set in the application. For AF_INET sockets, the value cannot exceed the number of connections allowed by the installed TCP/IP.

Rationale:

The specification states that an implementation may limit the length of a socket's listen queue, and that this limit may be imposed if the setting of the *backlog* argument exceeds an implementation-dependent maximum value.

Reference:

X/Open CAE Specification, Networking Services, Issue 4, Chapter 8, Sockets Interfaces, *listen()*.

6.3 Reference Manual Pages

6.3.1 Error Numbers

Question 3: Which of the following optional errors listed in the Reference Manual Pages are detected in the circumstances specified?

Response:

Function	Error	Detected
accept()	ENOMEM	No
	ENOBUFS	Yes
	ENOSR	No
	EPROTO	No
bind()	EINVAL	Yes
	EISCONN	No
	ENAMETOOLONG	No
	ENOBUFS	Yes
	ENOSR	No
connect()	EADDRINUSE	No
	ECONNRESET	No
	EHOSTUNREACH	No
	EINVAL	Yes
	ENAMETOOLONG	No
	ENETDOWN	No
	ENOBUFS	No
	ENOSR	Yes
	EOPNOTSUPP	
fgetpost()	ESPIPE	No
fsetpost()	ESPIPE	No
ftell()	ESPIPE	No
getpeername()	ENOBUFS	Yes
	ENOSR	No
getsockname()	EINVAL	No
	ENOBUFS	Yes
	ENOSR	No
getsockopt()	EINVAL	Yes
	ENOBUFS	Yes
	ENOSR	No
listen()	EINVAL	Yes
	ENOBUFS	Yes
recv()	EINVAL	Yes
	ENOBUFS	Yes
	ENOSR	No

Function	Error	Detected
recvfrom()	EIO	No
	ENOBUFS	Yes
	ENOMEM	No
	ENOSR	No
recvmsg()	EINVAL	Yes
	EIO	No
	ENOBUFS	Yes
	ENOMEMS	No
	ENOSR	No
send()	EINETDOWN	No
	EHOSTUNREACH	No
	ENOBUFS	Yes
	ENOSR	No
	EIO	No
sendmsg()	EDESTADDRREQ	No
	EHOSTUNREACH	No
	EINVAL	Yes
	EIO	No
	EISCONN	No
	ENAMETOOLONG	No
	ENETDOWN	No
	ENETUNREACH	No
	ENOBUFS	Yes
	ENOMEM	No
	ENOSR	
sendto()	DESTADDDRREQ	No
	EHOSTUNREACH	No
	EINVAL	Yes
	EIO	No
	EISCONN	No
	ENAMETOOLONG	No
	ENETDOWN	No
	ENETUNREACH	No
	ENOBUFS	Yes
	ENOMEM	No
	ENOSR	No
setsockopt()	ENOMEM	No
	ENOBUFS	Yes
	ENOSR	No
shutdown()	ENOBUFS	Yes
	ENOSR	No

Function	**Error**	**Detected**
socket()	ENOBUFS	Yes
	EMNOMEM	No
	ENOSR	No
socketpair()	EACCES	Yes
	EMNOMEM	No
	ENOBUFS	Yes
	ENOSR	No

Rationale:

Each of the above error conditions is marked as optional in the Reference Manual Pages and an implementation may return this error in the circumstances specified or may not provide the error indication.

Reference:

X/Open CAE Specification, Networking Services, Issue 4, Chapter 8, Sockets Interfaces, ERRORS.

XPG4 Internationalized Terminal Interfaces Profile

X/Open Conformance Statement
(Revised December 1995)

Type: XPG4 Component

Component Name: XPG4 Internationalized Terminal Interfaces

Completed by: International Business Machines Corporation
(name and organization)

on: February 4, 1998
(date)

Internationalized Terminal Interfaces

Product Identification

Product Identification	OS/390
Version/Release No.	Version 2 Release 4 or later

If you do not supply this component yourself, please identify below the supplier you reference:

With:

OS/390 V2R4 or later Security Server
OS/390 V2R4 or later C/C++ Compiler

Indicator of Compliance

None defined for this component.

Environment Specification

a. Testing Environment:

   See Attachment C.
   See Attachment F, OEBRAND1.

b. Binary-compatible Family:

   IBM System/390 Processors that support OS/390 Version 1 Release 3 or later.
   See Attachment B.

c. Portability Environment:

   The environment contains an XPG4 Internationalized System Calls and Libraries Extended Branded component.

Temporary Waivers

None.

7.1 Product Scope

7.1.1 Type of Conformance

Question 1: What is the scope of conformance of this product?

Response:

Transitional Conformance.

No known functional differences are noted between the bounded product and the X/Open Terminal Interfaces component specifications.

Rationale:

The conformance requirements for this components have a built-in time dependency. At the time of publication of the XPG4 Components Definition, Version 2, the conformance requirements for XPG4 Internationalized Terminal Interfaces involves “transitional conformance” plus a commitment to move to “hard conformance” within a maximum of three months after it becomes applicable. After this three-month period, hard conformance becomes mandatory.

Reference:

XPG4 Internationalized Terminal Interfaces Component Definition.

7.1.2 Coded Character Sets

Question 2: Which coded character sets are supported by the **chtype** ***data type?***

Response:

Our Implementation is based on the EBCDIC code page and supports several octet-based code sets, such as:

IBM-037 IBM-273 IBM-277 IBM-278 IBM-280 IBM-284 IBM-285 IBM-297 IBM-500 IBM-871 IBM-875 IBM-939 IBM-1027 IBM-1047

Rationale:

An implementation that claims XPG4 BASE, XPG4 Base 95 or XPG4 UNIX conformance and XPG4 Internationalized Terminal Interfaces conformance must support at least the ISO 8859-1 coded character set within the **chtype** data type. Support for other coded character sets is implementation-defined.

Reference:

X/Open CAE Specification, X/Open Curses, Issue 4, Section 1.2, Conformance.

7.1.3 Character Attributes

Question 3: Which character attributes are supported by the implementation?

Response:

A_ALTCHARSET	Yes
A_HORIZONTAL	No
A_LEFT	No
A_LOW	No
A_RIGHT	No
A_TOP	No
A_VERTICAL	No

Rationale:

An implementation that claims XPG4 BASE, XPG4 Base 95 conformance and XPG4 Internationalized Terminal Interfaces conformance must support at least the character attributes: A_BLINK, A_BOLD, A_DIM, A_REVERSE, A_STANDOUT, and A_UNDERLINE. Support for other character attributes listed above, is implementation-defined.

Reference:

X/Open CAE Specification, X/Open Curses, Issue 4, Chapter 5, Headers

7.2 Supported Terminals

Question 4: Which of the following terminal types are supported by the implementation (if any)?

Response:

Synchronous—No
Networked Asynchronous—No
Nonstandard Asynchronous—No
A character cannot be transmitted by a single keystroke only in blocks—N/A
The **refresh()** routine must redraw the entire screen contents in order to perform an update—N/A
It is not possible to disable echo—N/A
There are other limitations defined below—N/A

Rationale:

The General Terminal Interface described in System Interface Definitions, Issue 4, Version 2, and the Curses definition defined in X/Open Curses, Issue 4, are provided to control terminals connected to asynchronous communication ports. They may also be used to control synchronous, networked synchronous, or non-standard directly-connected asynchronous terminals, subject to possible implementation-defined limitations.

Reference:

X/Open CAE Specification, System Interface Definitions, Issue 4, Version 2, Chapter 9, General Terminal Interface.

X/Open CAE Specification, X/Open Curses, Issue 4, Section 3.9, Synchronous and Networked Asynchronous Terminals.

7.3 Terminfo Source File Limits

Question 5: What limits does the implementation support for a terminfo source file?

Response:

Length of a line—1024 bytes
Length of a terminal alias—14 bytes
Length of a terminal model name—128 bytes
Length of a single field—128 bytes
Length of a string value—2048 bytes
Length of a string representing a numeric value—99 digits
Magnitude of a numeric value—0–32767

Rationale:

X/Open Curses, Issue 4, specifies that a conformant implementation must declare its actual limits for the above items and defines minimum values that the implementation must support.

Reference:

X/Open CAE Specification, X/Open Curses, Issue 4, Section 6.1.1, Minimum Guaranteed Limits.

Attachments

Attachment A

Software Environment

In addition to the steps described in the VSX documentation, this attachment describes the OS/390-specific steps that were required for VSX testing.

User IDs were added for the following test IDs:

```
vsx0::240:400::/vsx/usr/TET/vsx4::/bin/sh
vsx1::241:401::/vsx/usr/vsx1::/bin/sh
vsx2::242:402::/vsx/usr/vsx2::/bin/sh
```

Group IDs were added for the following test IDs:

```
vsxg0::400:vsx0
```

Note: NGROUPMAX is 300, therefore, vsx0 is in each of the 299 supplementary groups that start with A0*.

```
A0001::551:vsx0
    through
A0299::849:vsx0

vsxg1::401:vsx1
vsxg2::402:vsx2
vsxg3::403
vsxg4::404
vsxg5::405
vsxg6::406
```

An HFS (hierarchical file system) dataset was created with 700 cylinders for the primary extent and 100 cylinders for the secondary extent on a 3390-3 DASD using the OpenEdition ISHELL File_systems pull-down menu, option 2. An HFS dataset with 1 cylinder for the primary extent and no secondary extents was also created on a 3390-3 DASD for the spare filesystem.

The initial permissions of a newly created filesystem are 700, for security reasons. Therefore, the following steps are required only one time after creating the 2 filesystems:

```
  use ISHELL to mount the filesystem on /vsx
  From a superuser shell, issue:
     chmod 777 /vsx/filesystem_name
  unmount the filesystem and repeat with the second.
```

The 700 cylinder HFS was mounted on /vsx.

The following directories were created:

```
/vsx/usr
/vsx/usr/TET
/vsx/usr/TET/vsx4
/vsx/usr/lib/nls/locale
/vsx/usr/vsx1
/vsx/usr/vsx2
```

A userid alias table was created to enable mixed-case user names. The file /vsx/aliases was created by superuser with permissions 755 and the following contents:

```
/* MVS aliases */
VSX0     vsx0
VSX1     vsx1
```

The following commands were issued from a superuser shell:

```
chown -R vsx0:vsxg0 /vsx/usr
chown    vsx1:vsxg1 /vsx/usr/vsx1
chown    vsx2:vsxg2 /vsx/usr/vsx2
```

TET 1.10 was unwound in /vsx/usr/TET with the following command:

```
pax -rzf /vsx/TET.pax.Z
```

and two changes were necessary. The first change was to TET/src/posix_c/tcc/tool.c. An **open()** is issued in a way that is not POSIX.1 compliant. There are four instances and each have been changed from:

```
 rc = open(LOCK_FILE,O_CREAT|O_EXCL,
                     (S_IRUSR|S_IXUSR|S_IRGRP|S_IXGRP|S_IROTH|S_IXOTH))
```

to

```
 rc = open(LOCK_FILE,O_CREAT|O_EXCL|O_RDONLY,
                     (S_IRUSR|S_IXUSR|S_IRGRP|S_IXGRP|S_IROTH|S_IXOTH))
```

The second change was to TET/src/posix_c/api/tet_exec.c. If getenv (TET_CONFIG) returns a non-NULL character pointer, **tet_exec()** expects to be able to subtract the length of the variable name plus one byte to obtain a string in the form "TET_CONFIG=value" which can be used to populate an environment variable array.

This use of **getenv()** is not valid on our platform and the environment variables do not get set correctly. The array **tet_exec()** builds has strings which do not contain equal signs and causes our **exec()** functions to fail.

Here are the changes:

```
/*  define the following function   */
char *env_build(char *var, char *val) {
  char *ret;

  if ((ret = (char*) malloc(strlen(var)+strlen(val)+2)) != NULL) {
    strcpy(ret, var);
    strcat(ret, "=");
    strcat(ret, val);
  }
  return ret;
}
.
.
.

/* change occurrences of   */
cp -= sizeof(TET_CONFIG);
   /*  to   */
cp = env_build(TET_CONFIG, cp);
```

The Makefile for TET was modified for our implementation and TET was built.

The archived VSX4.3.5 test suite was loaded into the filesystem.

The test suite was unarchived in /vsx/usr/TET/vsx4 with the following command:

```
pax -o from=ISO8859-1,to=IBM-1047 -rzf /vsx/vsx435.cpio.Z
```

The VSX4.3.5C patch was unarchived with a similiar command from the superuser id. The superuser id was used to insure that any files without write permission would be overwritten.

Profile (.profile) files were created for each tester user. The following environment variables were added to vsx0's .profile:

```
export PATH="/bin:$HOME/BIN/:$HOME/../bin:/vsx/bin"
export TET_EXECUTE=$HOME/TESTROOT
export LOCPATH=/vsx/usr/lib/nls/locale
export _CEE_RUNOPTS='errcount(0)'
```

Note:

> The LOCPATH environment variable is required to identify the directory for the setlocale() function.
>
> The _CEE_RUNOPTS environment variable initializes the runtime option, ERRCOUNT().

A character special file was created by issuing the following command from a superuser shell:

```
# /usr/sbin/mknod  /vsx/nxio.charspec c 1 65535
```

This file was then chown'ed to vsx0 and vsxg0, and chmod'ed to 666.

VSX pseudo-languages VSX4L0, VSX4L1, VSX4L2, VSX4L3@dict and VSX4L3 were configured.

The following steps were executed by vsx0 to install the psuedo-languages:

```
  $ cp  /SRC/INC/ctrlebcdic.h   /SRC/INC/ctrlcodes.h
  $ cp  /SRC/INC/pslebcdic.h   /SRC/INC/pslcodes.h
  $ cd  /SUPPORT/psldefs
  $ ed Makefile
  966
  /¬CHARSET
  CHARSET=<tab>ascii
  s/ascii/ebcdic/                            /* see Note 1 below   */
  /¬INSTDIR
  INSTDIR=<tab>./
  s&;/&/vsx/usr/lib/nls/locale/&;
  1,$s/¬<tab>localedef/<tab>-localedef/      /* see Note 2 below   */
  wq
  1005
  $ make
```

Note 1: This will point to EBCDIC charmap files instead of ASCII charmap files.

Note 2: The **localedef** utility was run with the - (dash) to insure that make ignores a nonzero exit value. Localedef may return nonzero and issue many warning messages if a locale source file does not contain all of the character classifications.

VSX4.3.5C was then configured (config.sh was executed by vsx0 in his home directory).

SRC/userintf.c was modified to meet our system requirements.

The SRC/vsxparams file was edited, changing TSORT to cat.

The SRC/vsxconfig.h file was edited, changing NSIG from -1 to 39.

Since OpenEdition supports statefully encoded multibyte character sets, the following copy was issued:

```
cp    /SRC/wc_nosup.cfg  /SRC/wchars.cfg
```

This was required since the test suite tests only non-state dependent character encodings and our multi-byte characters are statefully encoded.

The SRC/common/vtools/y.tab.c file contains ASCII dependencies and needed to be rebuilt to execute on an EBCDIC platform. We removed the y.tab.c file and this caused it to be rebuilt during the install phase.

The following steps were required since OpenEdition does not support the su utility as used in the highest level makefile of vsx0's home directory:

```
chmod 755 TESTROOT
chmod g+s TESTROOT
install.sh            /* run by vsx0  */

# the spare filesystem was mounted by su
# and it was made the current working directory

   vsx0/SRC/install/scripts/filldisc.sh     /* run by su    */

# su changed current working directory to vsx0's home directory
# and the spare filesystem was unmounted

chown 0 SRC/BIN/chmog      /* run by su    */
chmod u+s SRC/BIN/chmog    /* run by su    */
```

The following modification to tetbuild.cfg was necessary:

```
       change
    CC=/bin/c89
       to
    CC=/vsx/bin/c89.preproc
```

/vsx/bin/c89.preproc is a preprocessor filter that is required because the C/370 preprocessor generates [pragma filetag('code-page-name')[as the first line of output. This is used by the C/370 compiler as identification of the code-page of the data. This is extra information required for subsequent compilation phases.

Here are the contents of /vsx/bin/c89.preproc:

```
$ cat /vsx/bin/c89.preproc
# invoke c89 and eliminate "#pragma" statements from stdout introduced
# by compiler when -E is used to produce preprocessor output.

file=${TMPDIR:-/tmp}/$$.out
/bin/c89 $* > $file
rc=$?
grep -v '??=pragma' $file
rm $file
exit $rc
```

The execution parameters file, TESTROOT/tetexec.cfg, was customized. The VSX_TTYNAME for the tester was checked using the **tty** shell command. The tetexec.cfg file was modified to match the tty. The VSX_NAME is added to identify the test reference number.

The test suite was executed.

Attachment B

Environment Specification

Binary-compatible Family:

- All models of the S/390 Parallel Enterprise Servers or S/390 Parallel Transaction Servers (IBM 9672)
- All models of the S/390 Parallel Enterprise Servers - Generation 3(TM)
- All models of the S/390 Multiprise 2000 (TM)
- All models of the IBM ES/9000 (TM) Processor Unit 9021, the 9121, or the 9221
- An IBM ES/3090-9000T(TM) processor (Models 15T, 17T, 18T, 25T, 28T) that supports IBM Enterprise Systems Architecture/370 (TM) (ESA/370(TM)) and can have optional ESA/390 facilities.
- An IBM ES/3090 (TM) Model E, S, J, or JH processor at the appropriate engineering change (EC) level that supports the IBM Enterprise Systems Architecture/370(TM).
- An IBM ES/4381 (TM) Model Group 90E, 91E or 92E processor that supports the IBM Enterprise Systems Architecture/370.
- Enhanced LPAR mode operation is supported on all PR/SM(TM)-capable IBM processors in hardware configurations of two or more CPs with the exception of the ES/9000 Processor Unit 9221 Model 200.
- PC Server System/390 or RS/6000 with S/390 Server-on-Board.
- Coupling Facility - A production coupling facility can be an IBM 9674 running the coupling facility control code in a PR/SM LPAR. The coupling facility control code can also run in a 9021 711-based model, or a S/390 Parallel Enterprise Server or S/390 Parallel Transaction Server (IBM 9672) in a PR/SM LPAR. For additional information, see IBM Hardware Announcement 194-082, dated April 6, 1994 (RFA 22416 - Coupling Facility) and IBM Hardware Announcement 194-281, dated September 13, 1994.
- Coupling Facility Channels: Coupling facility channels (also referred to as coupling links), are high bandwidth fiber optic links that provide high speed connectivity between the coupling facility control code and the MVS/ESA or OS/390 systems running in CEC's (also referred to as CPC's) which use the services of the coupling facility control code. These channels are supported on all 9021 711-based models, 9121 511-based models, and the S/390 Parallel Enterprise Server or S/390 Parallel Transaction Server (IBM 9672)
- Sysplex Timer(R): The Sysplex Timer is required to synchronize the time-of-day (TOD) clocks in all the CPCs attached to the coupling facility.

Attachment C

Required Service to OS/390 Release 4 for a UNIX-Branded System

```
LIST APARS  FORFMID(HBB6603,JBB6604,
                     HMWL810,
                     JMWL81B,
                     JMWL81D,
                     JMWL85H,
                     HDZ11D0,
                     JDZ11DB,
                     HLB4701,
                     JLB4702,
                     JLB4703,
                     HTV4721,
                     JTV4722,
                     JTV4723,
                     HLC4731,
                     HCKVB00,
                     JCKVB03,
                     JCKVB05,
                     HTCP320,
                     JTCP32C,
                     JTCP322,
                     JTCP323,
                     JTCP325,
                     JTCP326,
                     JTCP329,
                     JTCP32H,
                     JTCP32G,
                     JTCP32N,
                     HOT1160)
```

```
AQ04250   TYPE            = APAR
          STATUS          = REC  APP
          FMID            = HMWL810
          DATE/TIME REC   = 97.231  10:06:14
                    INS   = 97.231  14:41:10
          LASTSUP         = UQ09322
          PRE    VER(001) = UQ05048   UQ06205   UQ07730
          REQ    VER(001) = AQ04251
          MAC             = CEECAA
          MOD             = CEEHDSP   CEEHGOTO  CEEVASTK  CEEVGTSI

AQ04251   TYPE            = APAR
          STATUS          = REC  APP
          FMID            = HMWL810
          DATE/TIME REC   = 97.231  10:06:14
                    INS   = 97.231  14:41:10
          LASTSUP         = UQ09281
          SUPBY(IN SYSMD) = BQ04251
          REQ    VER(001) = AQ04250
          MOD             = EDC40217  EDC403AB  EDC403AC

AQ04509   TYPE            = APAR
          STATUS          = REC  APP
          FMID            = HMWL810
          DATE/TIME REC   = 97.158  11:32:39
                    INS   = 97.159  18:03:51
          LASTSUP         = UQ06647
          MOD             = EDC4$0E0  EDC4$0E3  EDC4$0E6  EDC4$0E8  EDC4$0FB  E

AQ05944   TYPE            = APAR
          STATUS          = REC  APP
```

```
                   FMID            = HMWL810
                   DATE/TIME REC   = 97.218  14:13:12
                             INS   = 97.218  14:55:35
                   SOURCEID        = SERV001
                   MOD             = EDC40098

         AQ07134   TYPE            = APAR
                   STATUS          = REC  APP
                   FMID            = HMWL810
                   DATE/TIME REC   = 97.218  15:56:05
                             INS   = 97.218  15:58:23
                   MOD             = EDC4001F

         AQ07313   TYPE            = APAR
                   STATUS          = REC  APP
                   FMID            = JMWL85H
                   DATE/TIME REC   = 97.238  13:07:37
                             INS   = 97.238  13:31:14
                   PRE    VER(001) = UQ08655
                   IFREQ           = BQ07313
                   HFS             = EDC4H001  EDC4H013


         AQ07314   TYPE            = APAR
                   STATUS          = REC  APP
                   FMID            = JMWL85H
                   DATE/TIME REC   = 97.238  13:07:37
                             INS   = 97.238  13:31:00
                   PRE    VER(001) = UQ08506
                   IFREQ           = BQ07314
                   HFS             = EDC4H00B

         AQ07315   TYPE            = APAR
                   STATUS          = REC  APP
                   FMID            = JMWL85H
                   DATE/TIME REC   = 97.238  13:07:37
                             INS   = 97.238  13:31:00
                   PRE    VER(001) = UQ08570
                   IFREQ           = BQ07315
                   HFS             = EDC4H032

         AW25986   TYPE            = APAR
                   STATUS          = REC  APP
                   FMID            = HOT1160
                   JCLIN           = YES
                   DELLMOD         = YES
                   DATE/TIME REC   = 97.218  14:13:12
                             INS   = 97.218  14:55:36
                   SOURCEID        = SERV001
                   DLMOD           = FSUMSLDF
                   MOD             = FSUMXLDF

         AW27171   TYPE            = APAR
                   STATUS          = REC  APP
                   FMID            = HDZ11D0
                   REWORK          = 1997153
                   DATE/TIME REC   = 97.162  11:36:09
                             INS   = 97.162  11:37:41
                   LASTSUP         = UW38904
                   SUPING VER(001) = AW23308
                   MAC             = EXLVS     IDACB2    IFGEXLST

         AW28225   TYPE            = APAR
                   STATUS          = REC  APP
                   FMID            = JBB6604
                   DATE/TIME REC   = 97.218  14:13:13
                             INS   = 97.218  14:54:37
                   SOURCEID        = SERV001
                   MOD             = BPXPRNIC  BPXPRSPY

         AW28605   TYPE            = APAR
                   STATUS          = REC  APP
                   FMID            = JBB6604
                   DATE/TIME REC   = 97.218  14:13:13
                             INS   = 97.218  14:54:38
                   SOURCEID        = SERV001
                   PRE    VER(001) = UW40727
                   MOD             = BPXFSLIT

         AW28629   TYPE            = APAR
                   STATUS          = REC  APP
                   FMID            = JBB6604
```

```
          DATE/TIME REC   = 97.226  09:48:20
                    INS   = 97.231  14:28:12
          MOD             = BPXPRFK

AW28681   TYPE            = APAR
          STATUS          = REC  APP
          FMID            = HOT1160
          DATE/TIME REC   = 97.223  11:48:48
                    INS   = 97.240  09:33:12
          LASTSUP         = CW28681
          SUPBY(IN SYSMD) = BW28681
          PRE    VER(001) = UW39523
          MOD             = FSUMXMV   FSUMXSH

AW28814   TYPE            = APAR
          STATUS          = REC  APP
          FMID            = HDZ11D0
          DATE/TIME REC   = 97.240  11:46:49
                    INS   = 97.240  12:32:33
          MOD             = GFUGOPEN  GFUGRDWR  GFUGTRUN

BQ04251   TYPE            = APAR
          STATUS          = REC  APP
          FMID            = HMWL810
          DATE/TIME REC   = 97.247  15:23:20
                    INS   = 97.248  09:23:53
          LASTSUP         = UQ09281
          REQ    VER(001) = AQ04250
          SUPING VER(001) = AQ04251
          MOD             = EDC40217  EDC403AB  EDC403AC

BQ07313   TYPE            = APAR
          STATUS          = REC  APP
          FMID            = HMWL810
          DATE/TIME REC   = 97.238  13:27:49
                    INS   = 97.238  13:31:14
          PRE    VER(001) = UQ08636
          IFREQ           = AQ07313
          DATA            = EDC4H001  EDC4H013

BQ07314   TYPE            = APAR
          STATUS          = REC  APP
          FMID            = HMWL810
          DATE/TIME REC   = 97.238  13:27:49
                    INS   = 97.238  13:31:00
          PRE    VER(001) = UQ08503
          IFREQ           = AQ07314
          DATA            = EDC4H00B

BQ07315   TYPE            = APAR
          STATUS          = REC  APP
          FMID            = HMWL810
          DATE/TIME REC   = 97.238  13:27:49
                    INS   = 97.238  13:31:00
          PRE    VER(001) = UQ08567
          IFREQ           = AQ07315
          DATA            = EDC4H032

BW27883   TYPE            = APAR
          STATUS          = REC  APP
          FMID            = JBB6604
          DATE/TIME REC   = 97.218  14:13:13
                    INS   = 97.218  14:54:38
          LASTSUP         = UW40690
          SOURCEID        = SERV001
          PRE    VER(001) = UW38542   UW38622
          SUPING VER(001) = AW27883
          MOD             = BPXNSDLV  BPXNSIR1  BPXNSSIA

BW28593   TYPE            = APAR
          STATUS          = REC  APP
          FMID            = JBB6604
          DATE/TIME REC   = 97.218  14:13:13
                    INS   = 97.218  14:54:38
          SOURCEID        = SERV001
          PRE    VER(001) = UW38542
          SUPING VER(001) = AW28593
          MOD             = BPXFSRDW  BPXNSGNS

BW28681   TYPE            = APAR
          STATUS          = REC  APP
          FMID            = HOT1160
          DATE/TIME REC   = 97.240  11:05:42
```

```
                    INS   = 97.240  12:32:33
          LASTSUP         = CW28681
          PRE    VER(001) = UW39523
          SUPING VER(001) = AW28681
          MOD             = FSUMXMV   FSUMXSH

CW28681   TYPE            = APAR
          STATUS          = REC  APP
          FMID            = HOT1160
          DATE/TIME REC   = 97.241  16:41:38
                    INS   = 97.241  17:08:07
          PRE    VER(001) = UW39523
          SUPING VER(001) = AW28681   BW28681
          MOD             = FSUMXMV   FSUMXSH

DQ07760   TYPE            = APAR
          STATUS          = REC  APP
          FMID            = JTCP329
          DATE/TIME REC   = 97.322  13:07:14
                    INS   = 97.322  13:38:55
          LASTSUP         = UQ11945
          PRE    VER(001) = UQ07712   UQ09677   UQ10299   UQ11035
          SUPING VER(001) = AQ07760   BQ07760   CQ07760
          MOD             = EZBIPLIF  EZBPFACP  EZBPFGNM  EZBPFRED  EZBSKCIB  E
                            EZBTCRDG  EZBTCREQ  EZBTCSTR  EZBTCWRT  EZBTCWTE  E

KPFIX00   TYPE            = APAR
          STATUS          = REC  APP
          FMID            = JBB6604
          DATE/TIME REC   = 97.168  10:19:21
                    INS   = 97.168  10:20:51
          MOD             = IWMDNSRV
```

Attachment D

Software Environment

In addition to the steps described in the VSC documentation, this attachment describes the OS/390-specific steps that were required for VSC testing.

User IDs were added for the following test IDs:

```
vsc0::40:400::/vsc::/bin/sh
vsc1::41:400::/vsc/usr/vsc1::/bin/sh
vsc2::42:400::/vsc/usr/vsc2::/bin/sh
```

Group IDs were added for the following test IDs:

```
vscg0::1400:vsc0
vscg1::1401:vsc1
vscg2::1402:vsc2
vscg4::1404
```

Groups mail and uucpg were added for mailx and uu* tests.

Users uucp and nuucp were added for uu* tests.

An HFS (hierarchical file system) dataset was created with 700 cylinders for the primary extent and 100 cylinders for the secondary extent on a 3390-3 DASD using the OpenEdition ISHELL File_systems pull-down menu, option 2. Two additional HFS datasets with one cylinder for the primary extent and no secondary extents were also created on a 3390-3 DASD for the spare and the read-only file systems.

The initial permissions of a newly created filesystem are 700, for security reasons. Therefore, the following steps are required only one time after creating the two file systems:

```
use ISHELL to mount the filesystem on /vsc
  From a superuser shell, issue:
     chmod 777 /vsc/filesystem_name
  Also, the ownership needed to be changed on each filesystem with
  the following command:
      chown vsc0:vscg0 /vsc/filesystem_name
  unmount the filesystem and repeat with the others.
```

The 700 cylinder HFS was mounted on /vsc.

The following directories were created:

```
/vsc/usr
/vsc/usr/vsc0
/vsc/usr/vsc1
/vsc/usr/vsc2
```

The following commands were issued from a superuser shell:

```
chown -R vsc0:vscg0 /vsc/usr
chown    vsc1:vscg1 /vsc/usr/vsc1
chown    vsc2:vscg2 /vsc/usr/vsc2
```

Profile (.profile) was created for vsc0. The following environment variables were added to vsc0's .profile:

```
export PATH="/bin:$HOME/BIN/:$HOME/../bin"
export MAKESTARTUP=/vsc/usr/vsc0/startup.mk
export NLSPATH=/usr/lib/nls/msg/C/%N
export MANPATH=/usr/man/C
export TET_NSIG=39
export MAILRC=/vsc/mailrc
```

The MAKESTARTUP environment variable is required to identify the make startup file. This file will override the default rules and flags defined in the startup.mk file that is provided by OpenEdition. The following line was added to the end of the startup.mk:

```
.POSIX:
```

The NLSPATH environment variable is used by the localedef utility. The MANPATH environment variable is used by the man utility.

The following commands were issued to make sure that the TETROOT is owned by the tester (vsc0):

```
cd /vsc
mkdir VSC4.1.5
chown vsc0:vscg0 VSC4.1.5
```

The MAILRC environment variable tells mailx where the mailrc file is located. Here are the contents of this file:

```
$ cat vs0/mailrc
set sendmail=/usr/lib/tsmail
```

The following commands were issued to create directories to be used as mount points for the spare and read-only filesystems:

```
mkdir /vsc/spare /vsc/ronly
chown vsc0:vscg0 /vsc/spare /vsc/ronly
```

The archived VSC4.1.5 test suite was copied into the filesystem into **/vsc/vsc415.cpio.Z**.

The test suite was unarchived in /vsc with the following command:

```
cd /vsc
pax -o from=ISO8859-1,to=IBM-1047 -rzf /vsc/vsc415.cpio.Z
```

The following files were created:

```
/vsc/iconvtest
   the contents of this file are:
   $ cat iconvtest
   this is a test
   for the iconv utility
   0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ.-_,\
   abcdefghijklmnopqrstuvwxyz
```

The character special file was created by a superuser with the following command:

```
/usr/sbin/mknod /vsc/csf c 100 101
```

All files in /vsc should be owned by vsc0, so the following was issued by a superuser:

```
chown vsc0:vscg0 /vsc/*
```

The following steps were required to build the vsc4.1.5 test suite:

```
$ cd /vsc/V*5
$ . SOURCE_ME
```

Change the following TET module:

```
$TET_ROOT/src/posix_c/tcc/tool.c
```

An **open()** is issued in a way that is not POSIX.1 compliant. There are four instances and each have been changed from:

```
 rc = open(LOCK_FILE,O_CREAT|O_EXCL,
                    (S_IRUSR|S_IXUSR|S_IRGRP|S_IXGRP|S_IROTH|S_IXOTH))
```

to

```
 rc = open(LOCK_FILE,O_CREAT|O_EXCL|O_RDONLY,
                    (S_IRUSR|S_IXUSR|S_IRGRP|S_IXGRP|S_IROTH|S_IXOTH))
```

There were files in expect/tcl that required changes.

Following is a diff of the changed source with the original:

```
$ diff -b $TET_ROOT/vsc/Src/Interact/expect/exp_select.c.orig
 $TET_ROOT/vsc/Src/Interact/expect/exp_select.c
10a11,12
> #define _XOPEN_SOURCE_EXTENDED 1 /* jcp 11/22/94 */
> #define _OE_SOCKETS /* jcp 12/01/94 */
165a168,169
>              t = anytime;;  /* jcp  12/15  */
>              t->tv_sec = 15;  /* jcp 12/15 */
$

$ diff -b $TET_ROOT/vsc/Src/Interact/expect/pty_termios.c.orig
 $TET_ROOT/vsc/Src/Interact/expect/pty_termios.c
9a10,15
> /* change history:
>     12/08/94 John Pfuntner, pfuntner@vnet.ibm.com, change flag: "jcp"
>       Changed code to go after master/slave ptys that OpenEdition MVS
>       define: /dev/.pt.typNNNN.
> */
>
111a118
> /* jcp: original code:
113a121,123
> */
> static char   master_name[40]; /* jcp */
> static char   slave_name [40]; /* jcp */
482a493,505
>
>   /* jcp... */
>   {
>     int loop;
>     for (loop=0; master < 0; loop++) {
```

```
>       sprintf(master_name, "/dev/ptyp%04d", loop);
>       if (stat(master_name, stat_buf); < 0) break;
>       sprintf(slave_name,  "/dev/ttyp%04d", loop);
>       master = exp_pty_test(master_name, slave_name, loop%10, loop/10);
>     }
>   }
>   /* ...jcp */
>
$


$ diff -b $TET_ROOT/vsc/Src/Interact/tcl/tclParse.c
 $TET_ROOT/vsc/Src/Interact/tcl/tclParse.c.orig

29,41d28
< /* Change history:
<      12/08/94: John Pfuntner (pfuntner@vnet.ibm.com), change flag: "jcp"
<        Restructured the "tclTypeTable" array for the EBCDIC character
<        set, so that the expect utility might be used on OpenEdition
<        MVS.  I rearranged the element values and associated element 0
<        with the NUL character (0x00).  This is a change from the
<        original array that used the first 128 elements for (negative
<        characters), placing 0x00 at the 128th element.  Because I
<        associated element 0 with NUL, a change is also necessary to
<        the CHAR_TYPE macro in tcl/tclInt.h, the only part that
<        apparently references this array.
< */
<
49,53d35
<      jcp: this comment about "negative characters" is no longer
<           true, see my comment above in the change history.  Since
<           the default attribute for char types on OpenEdition MVS
<           is unsigned, there is no need to deal with "negative
<           characters".
57,314c39,102
< /* jcp... */
<     TCL_COMMAND_END,   /* NUL */
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_SPACE,         /* HT */
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_SPACE,         /* VT */
<     TCL_SPACE,         /* FF */
<     TCL_SPACE,         /* CR */
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_COMMAND_END,   /* LF (actually, I'm using NL) */
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_SPACE,         /* SP */
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_DOLLAR,        /* $ */
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_COMMAND_END,   /* ; */
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_QUOTE,         /* " */
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_OPEN_BRACKET,  /* [ */
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_COMMAND_END,   /* ] */
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_OPEN_BRACE,    /* { */
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_CLOSE_BRACE,   /* } */
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_BACKSLASH,     /* \ */
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
<     TCL_NORMAL,
< /* ...jcp */
---
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_COMMAND_END,  TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_SPACE,        TCL_COMMAND_END,  TCL_SPACE,
>     TCL_SPACE,        TCL_SPACE,        TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_SPACE,        TCL_NORMAL,       TCL_QUOTE,        TCL_NORMAL,
>     TCL_DOLLAR,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_COMMAND_END,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,       TCL_NORMAL,
>     TCL_NORMAL,        TCL_NORMAL,        TCL_NORMAL,        TCL_OPEN_BRACKET,
>     TCL_BACKSLASH,     TCL_COMMAND_END,   TCL_NORMAL,        TCL_NORMAL,
>     TCL_NORMAL,        TCL_NORMAL,        TCL_NORMAL,        TCL_NORMAL,
>     TCL_NORMAL,        TCL_NORMAL,        TCL_NORMAL,        TCL_NORMAL,
>     TCL_NORMAL,        TCL_NORMAL,        TCL_NORMAL,        TCL_NORMAL,
>     TCL_NORMAL,        TCL_NORMAL,        TCL_NORMAL,        TCL_NORMAL,
>     TCL_NORMAL,        TCL_NORMAL,        TCL_NORMAL,        TCL_NORMAL,
>     TCL_NORMAL,        TCL_NORMAL,        TCL_NORMAL,        TCL_NORMAL,
>     TCL_NORMAL,        TCL_NORMAL,        TCL_NORMAL,        TCL_OPEN_BRACE,
>     TCL_NORMAL,        TCL_CLOSE_BRACE,   TCL_NORMAL,        TCL_NORMAL,


$ diff -b $TET_ROOT/vsc/Src/Interact/tcl/tclInt.h
 $TET_ROOT/vsc/Src/Interact/tcl/tclInt.h.orig
28,35d27
< /* Change history:
<      12/08/94: John Pfuntner (pfuntner@vnet.ibm.com), change flag: "jcp"
<        Restructured the "tclTypeTable" array in tclParse.c (see the
<        change history in that file for more information on why) so
<        I needed to change the CHAR_TYPE macro in this header file to
<        access the array without adding 128 to the initial index.
< */
<
677,678c669
< /* jcp, original: #define CHAR_TYPE(c) (tclTypeTable+128)[c] */
< #define CHAR_TYPE(c) (tclTypeTable)[c] /* jcp */
---
> #define CHAR_TYPE(c) (tclTypeTable+128)[c]
$

$ diff -b $TET_ROOT/vsc/Src/Interact/tcl/configure
 $TET_ROOT/vsc/Src/Interact/tcl/configure.orig
299d298
< #include <sys_time.h> /* jcp */
308d306
< struct timeval T; /* jcp */
310,312c308
< /*******extern char gettimeofday(); gettimeofday();   *******/
< extern int gettimeofday(struct timeval *, void *); /* jcp */
< gettimeofday(&T;,; NULL); /* jcp */
---
> extern char gettimeofday(); gettimeofday();
625d620
< /******** changed sys/time to sys_time----gettimeofday()  ATT******/
628c623
< #include <sys_time.h>
---
> #include <sys/time.h>
$

$ diff -b $TET_ROOT/vsc/Src/Interact/expect/configure
 $TET_ROOT/vsc/Src/Interact/expect/configure.orig
 470d469
 < ac_cv_prog_CPP="$CPP"   #required since CPP was losing it's value    att
 $
```

```
The $TET_ROOT/vsc/Lib/termin.exp file needed to be changed;
there were several ASCII dependencies.  Follows is a diff which
shows the changes:

$ diff -b $TET_ROOT/vsc/Lib/termin.exp
 $TET_ROOT/vsc/Lib/termin.exp.orig

3c3
< # $Header: /usr3/vsc/vsc/Lib/RCS/termin.exp,v 2.0 1995/09/23 22:07:09 tbr Rel $
---
> # $Header: /usr3/vsc/vsc/Lib/RCS/termin.exp,v 3.0 1995/10/11 03:31:32 tbr Rel $
37a38,40
> # Revision 3.0  1995/10/11  03:31:32  tbr
> # Branch point for Release 4.1.5
> #
112,124c115,127
<         "A" { return "\x01" } "B" { return "\x02" }
<         "C" { return "\x03" } "D" { return "\x37" }
<         "E" { return "\x2d" } "F" { return "\x2e" }
<         "G" { return "\x2f" } "H" { return "\x16" }
<         "I" { return "\x05" } "J" { return "\x15" }
<         "K" { return "\x0b" } "L" { return "\x0c" }
<         "M" { return "\x0d" } "N" { return "\x0e" }
<         "O" { return "\x0f" } "P" { return "\x10" }
<         "Q" { return "\x11" } "R" { return "\x12" }
<         "S" { return "\x13" } "T" { return "\x3c" }
<         "U" { return "\x3d" } "V" { return "\x32" }
<         "W" { return "\x26" } "X" { return "\x18" }
<         "Y" { return "\x19" } "Z" { return "\x3f" }
---
>         $ControlA { return "\x01" } $ControlB { return "\x02" }
>         $ControlC { return "\x03" } $ControlD { return "\x37" }
>         $ControlE { return "\x2d" } $ControlF { return "\x2e" }
>         $ControlG { return "\x2f" } $ControlH { return "\x16" }
>         $ControlI { return "\x05" } $ControlJ { return "\x15" }
>         $ControlK { return "\x0b" } $ControlL { return "\x0c" }
>         $ControlM { return "\x0d" } $ControlN { return "\x0e" }
>         $ControlO { return "\x0f" } $ControlP { return "\x10" }
>         $ControlQ { return "\x11" } $ControlR { return "\x12" }
>         $ControlS { return "\x13" } $ControlT { return "\x3c" }
>         $ControlU { return "\x3d" } $ControlV { return "\x32" }
>         $ControlW { return "\x26" } $ControlX { return "\x18" }
>         $ControlY { return "\x19" } $ControlZ { return "\x3f" }
```

The $TET_ROOT/vsc/Src/Shell/termin.tcl file needed to be changed; there were several ASCII dependencies. Follows is a diff which shows the changes:

```
$ diff -b $TET_ROOT/vsc/Src/Shell/termin.tcl
 $TET_ROOT/vsc/Src/Shell/termin.tcl.orig

3c3
< #      SCCS:  @(#) termin.tcl  Version 4.1.4   (95/05/20)
---
> #      SCCS:  @(#) termin.tcl  Version 4.1.4   (95/06/16)
46,59d45
< # Set up table of EBCDIC control characters        # ibm c1
< set control(A) \x01;  set control(L) \x0c;    set control(W) \x26;
< set control(B) \x02;  set control(M) \x0d;    set control(X) \x18;
< set control(C) \x03;  set control(N) \x0e;    set control(Y) \x19;
< set control(D) \x37;  set control(O) \x0f;    set control(Z) \x3f;
< set control(E) \x2d;  set control(P) \x10;    set control(\[) \x27;
< set control(F) \x2e;  set control(Q) \x11;    set control(\\) \x1c;
< set control(G) \x2f;  set control(R) \x12;    set control(\]) \x1d;
< set control(H) \x16;  set control(S) \x13;    set control(⅛) \x1e;
< set control(I) \x05;  set control(T) \x3c;    set control(_) \x1f;
< set control(J) \x15;  set control(U) \x3d;    set control(?) \x07;
< set control(K) \x0b;  set control(V) \x32;
<                                                    # end of ibm c1
<
62,75d47
< # proc tocntl ch {
< #  set ch [string toupper $ch]
< #
< #  # Convert to a decimal value
< #  scan $ch %c chval
< #
< #  # Convert to control -- this is for ascii.
< #  set cntlval [expr $chval - 64]
< #
< #  set ret [format %c $cntlval]
< #  return $ret
< # }
<                                            # ibm c2 commented
<                                            # above, added:
77,78d48
<    global control
<
81,82c51,52
<  # Convert to control -- this is for ebcdic.
<    set cntlval $control($ch)
---
>    # Convert to a decimal value
>    scan $ch %c chval
84c54,58
<    return $cntlval
---
>    # Convert to control -- this is for ascii.
>    set cntlval [expr $chval - 64]
>
>    set ret [format %c $cntlval]
>    return $ret
86d59
<                                          # end of ibm c2
136c109
<                        sendit \047 <ESC>                   # ibm c3  was 033
---
>                        sendit \033 <ESC>
193c166
< set stty_init "-icrnl -echo -ixon erase '⅛h' kill '⅛u' intr '⅛c'"
---
> set stty_init "-echo -ixon erase '⅛h' kill '⅛u' intr '⅛c'"
```

Follows is a list of all of the additional files that were modified for the system under test:

```
$TET_ROOT/src/posix_sh/api/makefile  /* set signals */
$TET_ROOT/src/xpg3sh/api/makefile  /* set signals */
$TET_ROOT/src/posix_c/api/makefile /* set DEFINES */
$TET_ROOT/src/posix_c/tcc/makefile /* set DEFINES */
$TET_ROOT/src/posix_c/makefile     /* set DEFINES */
$TET_ROOT/buildexpect              /* set CPP and CFLAGS */
$TET_ROOT/vsc/Src/Interact/tcl/Makefile.in /*set prefix, CFLAGS */
$TET_ROOT/vsc/Src/Interact/expect/Makefile.in /* " " & SETUID   */
$TET_ROOT/vsc/privscript
$TET_ROOT/vsc/Src/ImplSpec/ClearMail.sh /* path        */
$TET_ROOT/vsc/Src/ImplSpec/ExecAsUser.c /* added chmod */

$TET_ROOT/vsc/tetexec.cfg was modified.  The ebcdic octal values
for control characters needed to be supplied.
Verified that the information for _L_SYS is correct.  (Note:
if this was configured/run on xa1, it may be CMN, it should
be J10)


$TET_ROOT/vsc/Src/ImplSpec/libexec.exp was modified.  The timing
variables were modified for our system.
```

The VSC4.1.5 suite was then built, by issuing the following command from vsc0's shell in TET_ROOT, after running SOURCE_ME:

```
export _C89_CCMODE=1
./Build
unset _C89_CCMODE
```

Note: _C89_CCMODE is required since some of the makefiles in expect/tcl are not POSIX.2 compliant. With this variable set, c89 will accept options and operands in any order.

Insure that the following daemons have been started:

From su shell:

```
/usr/sbin/cron &
/usr/sbin/inetd /tmp/inetd.conf
```

For remote uucp testing:

From su shell:

```
echo UUCP >>/usr/lib/cron/cron.allow
```

From uucp shell:

```
crontab </vsc/uucp.cron
```

On the remote system, oeaix8, issue the following command from a uucp shell:

```
crontab </u/oebrand/uucp.cron
```

The /vsc/uucp.cron file on J10 contains the following:

```
* * * * * /usr/lib/uucp/uuxqt -s J10 >/dev/null 2>&1>&1;
* * * * * /usr/lib/uucp/uucico -f -r 1 -s oeaix8 >/dev/null 2>&1;
```

The /u/oebrand/uucp.cron file on oeaix8 contains the following:

```
* * * * * /usr/lib/uucp/uucico -r 1 -s J10 >/dev/null 2>&1;
* * * * * /usr/lib/uucp/uuxqt -s oeaix8 >/dev/null 2>&1;
```

Before starting the test suite, verify that vsc0's mailbox is empty. Remove **/usr/mail/VSC0** if necessary.

The test suite was executed with the following command:

```
tcc -bec vsc all      /* without stdout or err re-directed  */
```

Attachment E

Software Environment

The BPXPRMxx parmlib member was customized with the following values:

```
MAXPROCSYS (256)
AXPROCUSER(25)
MAXUIDS(32)
MAXFILEPROC(3000)
MAXPTYS(1000)
MAXCPUTIME(2147483647)
FORKCOPY(COPY)
```

For more information, see the section on customizing the BPXPRMxx parmlib members in *OS/390 OpenEdition Planning*.

In addition to the steps described in the VSU documentation, this attachment describes the OS/390-specific steps that were required for VSU testing.

User IDs were added for the following test IDs:

```
vsu0::340:3400::/vsu/VSU4.1.0/CAPI/vsu::/bin/sh
vsu1::341:3401::/vsu/usr/vsu1::/bin/sh
vsu2::342:3402::/vsu/usr/vsu2::/bin/sh
```

Group IDs were added for the following test IDs:

```
vsug0::3400:vsu0
U0001::3551:vsu0
    through
U0031::3581:vsu0

vsug1::3401:vsu1
vsug2::3402:vsu2
```

An HFS (hierarchical file system) dataset was created with 700 cylinders for the primary extent and 100 cylinders for the secondary extent on a 3390-3 DASD using the OpenEdition ISHELL File_systems pull-down menu, option 2. An HFS dataset with 1 cylinder for the primary extent and no secondary extents was also created on a 3390-3 DASD for the spare filesystem.

The initial permissions of a newly created filesystem are 700, for security reasons. Therefore, the following steps are required only one time after creating the two filesystems:

```
  use ISHELL to mount the filesystem on /vsu
  From a superuser shell, issue:
     chmod 777 /vsu
  unmount the filesystem and repeat with the second.
```

The 700 cylinder HFS was mounted on /vsu.

The following directories were created:

```
/vsu/usr
/vsu/usr/vsu1
/vsu/usr/vsu2
```

The following commands were issued from a superuser shell:

```
chown -R vsu0:vsug0 /vsu/usr
chown    vsu1:vsug1 /vsu/usr/vsu1
chown    vsu2:vsug2 /vsu/usr/vsu2

# TET_ROOT is /vsu/VSU4.1.0/vsu.

Profile (.profile) files were created for each tester user.
The following environment variables were added to vsu0's .profile:
export TET_ROOT=/vsu/VSU4.1.0/vsu
export PATH=/vsu/bin:/bin:/usr/bin:$TET_ROOT/CAPI/BIN:$TET_ROOT/bin:/:.
export PS1=J10/$LOGNAME' ! $PWD> '
export _C89_CCMODE=1
export _C89_CSYSLIB=""
export _CEE_RUNOPTS='errcount(0),heap(72K)'
export TZ=EST5EDT
export MANPATH=/usr/man:$TET_ROOT/CAPI/MAN
export LOCPATH=$TET_ROOT/CAPI/SRC/LOCALE
export _C89_OPTIONS='-U_OPEN_DEFAULT,-DWA_NORMAL'
```

Note: The LOCPATH environment variable is required to identify the directory for the setlocale() function.

The _CEE_RUNOPTS environment variable initializes the runtime option, ERRCOUNT().

TET 1.10 was unwound in /vsu/VSU4.1.0/vsu with the following command:

```
pax -rzf /vsu/tet.pax.Z
```

and four changes were necessary.

The first change was to $TET_ROOT/src/posix_c/tcc/tool.c. An open() is issued in a way that is not POSIX.1 compliant. There are four instances and each have been changed from:

```
rc = open(LOCK_FILE,O_CREAT|O_EXCL,
                    (S_IRUSR|S_IXUSR|S_IRGRP|S_IXGRP|S_IROTH|S_IXOTH))
```

to

```
rc = open(LOCK_FILE,O_CREAT|O_EXCL|O_RDONLY,
                    (S_IRUSR|S_IXUSR|S_IRGRP|S_IXGRP|S_IROTH|S_IXOTH))
```

The second change was to $TET_ROOT/src/posix_c/api/tet_exec.c. If getenv("TET_CONFIG") returns a non-NULL character pointer, tet_exec() expects to be able to subtract the length of the variable name plus one byte to obtain a string in the form "TET_CONFIG=value" which can be used to populate an environment variable array.

This use of **getenv()** is not valid on our platform and the environment variables do not get set correctly. The array **tet_exec()** builds has strings which do not contain equal signs and causes our **exec()** functions to fail.

Here are the changes:

```
/*  define the following function   */
char *env_build(char *var, char *val) {
  char *ret;

  if ((ret = (char*) malloc(strlen(var)+strlen(val)+2)) != NULL) {
    strcpy(ret, var);
    strcat(ret, "=");
    strcat(ret, val);
  }
  return ret;
}
.
.
.

/* change occurrences of   */
cp -= sizeof(TET_CONFIG);
   /*  to   */
cp = env_build(TET_CONFIG, cp);
```

The third was in **startit.c** (see file for changes made). The fourth was in **scenario.c** (see file for changes made). I made a copy of the original file. We may be able to get rid of the 3rd and 4th changes if and when all test cases compile. Until that time we need these fixes so that tcc will not get a fatal error when it can't find an executable, and die.

The Makefile for TET in **$TET_ROOT/src/posix_c** was modified for our implementation and TET was built by issuing the **make** command in this directory.

The archived VSU4.1.0 test suite was loaded into the filesystem.

The test suite was unarchived in /vsu/VSU4.1.0/vsu with the following command:

```
pax -rzf /vsu/VSU4.1.0.pax.Z -o from=ISO8859-1,to=IBM-1047
```

The test suite patch D was unarchived in /vsu/VSU4.1.0/vsu with the following command:

```
pax -rzf /vsu/VSU4.1.0D.pax.Z -o from=ISO8859-1,to=IBM-1047
```

Made a copy of **/vsu/VSU4.0.2/vsu/CAPI/SRC/TOOLS/namespace/tree.c** and modified it because of message:

```
ERROR CBC3152 ./tree.c:803   A register array may only be used as the
operand to sizeof.
```

Our compiler does not support this usage.

Changed lines 745 and 789 as follows:

```
va_list p;  /* removed register attribute */
```

The **SRC/TOOLS/namespace/y.tab.c file**contains ASCII dependencies and needed to be rebuilt to execute on an EBCDIC platform. We removed the **y.tab.c** file and this caused it to be rebuilt during the install phase.

```
touch /usr/include/sys/param.h
chmod 744 /usr/include/sys/param.h
cp /usr/lib/libm.a /usr/lib/libxnet.a
chmod 777 /usr/lib/libxnet.a
touch /usr/include/sysdep.h
chmod 744 /usr/include/sysdep.h
cp /tmp/syslog.conf /etc/syslog.conf
touch /vsu/syslog.conf
chmod 700 /vsu/syslog.conf
cp /vsu/syslog.conf  /etc/syslog.conf
cp /usr/sbin/syslogd  /bin/syslogd
```

Created **/vsu/vsusyslog** as specified in **tetexec.cfg** from user vsu0 using the **touch** command.

Changed directory to $TET_ROOT/CAPI and executed “cmake install.”

Since our compiler outputs messages during compilation other than errors, we must suppress these messages for VSU. We have done this by creating a script to filter out these messages and modifying **tetbuild.cfg**.

```
Changed tetbuild.cfg
                     to     CC=/vsu/bin/c89
cat /vsu/bin/c89
# invoke c89 and eliminate the following:
#   1) "#pragma" statements from stdout introduced by compiler when -E
#      is used to produce preprocessor output
#   2) pre-link warning messsages relating to duplicate external
#      symbols.
out=${TMPDIR:-/vsu}/$$.out
/bin/c89 $* > $out 2>&1;
rc=$?
sed < $out '
        /??=pragma/d
        /FSUM3065 The PRELINK step ended with return code 4/d'
rm $out
exit $rc
```

We changed the premissions on /vsu/bin/c89 to 744 with chmod.

VSU4.1.0D was then configured by vsu0 in his home directory.

Made a copy of the original **tetexec.cfg file.** Modified the copy to reflect our implementataion. Made a backup copy called **tetexec.cfg.IBM.**

To establish some of the other configuration variables in **tetexec.cfg**, it is necessary to view several different datasets. This must be done before any run, branding or otherwise, as passwords and groups change dynamically on our systems. Therefore, to obtain the most recent snapshot of the group and password databases run program **/tmp/VSU4.0.2/vsu/CAPI/getvsuconfig.** This prints out some of the more volatile configuration variables which must be correct in tetexec.cfg immediately proceeding any branding runs. It is invoked from userid VSU0 in the following manner:

```
cd; ./getvsuconfig
```

Run checkconfig to verify configuration options are correctly set for our implementation. One error message was generated. Sent correspondence to vsu_support as follows regarding that message:

```
----------------------------------------------------------------
 From: SMTP4   --TPAVMPS2                   Date and time    08/22/96 21:08:02
 ========================================================================
 Received: from anchovy.aptest.com by vnet.IBM.COM (IBM VM SMTP V2R3) with TCP;
    Thu, 22 Aug 96 21:07:42 EDT
 Received: by anchovy.aptest.com (8.7.5)
 :id SAA26788; Thu, 22 Aug 1996 18:06:51 -0700 (PDT)
 Date: Thu, 22 Aug 1996 18:06:51 -0700 (PDT)
 From: tbr@aptest.com (Terry Rhodes)
 Message-Id: <199608230106.SAA26788@anchovy.aptest.com>
 Fencing: Life with an edge
 X-Mailer: Mail User's Shell (7.2.5 10/14/92)
 To: .
 Subject: VSU SR# 524 opened and response

 We recently received the enclosed support request from you via
 X/Open.  It was assigned the SR# given in the subject line above.
 It is important to reference this number in any future
 correspondence regarding this support request.

 Our response to your request follows.  If our response does not
 completely resolve your request please let us know as soon as
 possible.



 RESPONSE:

 These variables expect 3 fields because the generic UNIX definition
 of a protocols database entry is

      For each protocol a single line should be present with the
      following information:

           official-protocol-name protocol-number aliases

 These 3 fields correspond to the members of the protoent structure.

 If your implementation only uses 2 fields I believe that you
 should define these variables with an additional comma. For
 example

      VSU_PROTO1_ENTRY=ip,0,
      VSU_PROTO2_ENTRY=icmp,1,

 This should make both vsuconfig and the tests happy.



 ORIGINAL REQUEST:

 > Date: Thu, 22 Aug 1996 16:57:30 GMT
 > Reply-To: .
 > Subject: X/Open VSU4 support request 524
 > Date: Thu, 22 Aug 96 11:50:06 EDT
 > From: .
 > Subject: (vsu_support 524) checkconfig error messages
 >
 > When I ran checkconfig after updating my tetexec.cfg file
 > two error messages were produced indicating that our values
 > for VSU_PROTO_ENTRY1 and VSU_PROTO_ENTRY2 were incorrect because
 > we only used 2 comma separated fields but 3 fields are required.
 > Our TCPIP datasets contain only 2 fields.
 >
 > There do not seem to be any bad effects from this mismatch when we
 > run the build/execution of the VSU4 test suite.
 >
 > Is there anything we must do to correct this situation?  Or,
 > can we just ignore the error messages?
 >
----------------------------------------------------------------
```

Review the configuration checklist at the end of the chapter 5 and verify that we were ready to begin a clean of the VSU4 test suite.

The test suite was built and executed.

Attachment F

Hardware Environment

- The J10 processor. J10 is one (1) 6 cp squadron (9672) CEC in the middle of an EO8 (J00–J70).

  Installed Memory: 256MG
  Hardware System Area: 16MG
  Central Storage: 240MG
  Expanded Storage: 0MG

  Operating Mode: LPAR (2 LPAR zones)
    OEBRAND1(J10): 168MG
    OEBRAND2(J11): 72 MG

- The DASD used:

  Disk Controllers (3990-L03(1 used), 3390-Q06 (2 used)

  Disk Devices:

  +0A99-0A9F (3390 Model 3)
  +0D8A (3390 Model 3)
  +0DA2 (3390 Model 3)
  +0DA3 (3390 Model 3)
  +0A07 (330 Model K)
  +601c-601f (3390 Model 3)
  +6026 (3390 Model 3)
  +6027 (3390 Model 3)

- Terminal Controller: 3174–01L

Software Environment

Install all the elements of OS/390 Version 1 Release 2, including:

- OS/390 V1R2 Security Server
- OS/390 V1R2 C/C++ Compiler

The BPXPRMxx parmlib member was customized with the following values:

- MAXPROCSYS(256)
- MAXPROCUSER(25)
- MAXUIDS(32)
- MAXFILEPROC(3000)
- MAXPTYS(1000)
- MAXCPUTIME(2147483647)
- FORKCOPY(COPY)

For more information, see “Customizing the BPXPRMxx Parmlib Member” in *OS/390 OpenEdition Planning*.

Attachment G

Software Environment

In addition to the steps described in the VST documentation, this attachment describes the OS/390-specific steps that were required for VST testing.

All of the steps below were performed on BOTH the master and slave systems unless otherwise noted.

User IDs were added for the following test IDs:

```
vst0::134::/vst/usr/DTET2/vst::/bin/sh
```

Group IDs were added for the following test IDs:

```
vstg0::533:vst0
```

An HFS (hierarchical file system) dataset was created with 500 cylinders for the primary extent and 100 cylinders for the secondary extent on a 3390-3 DASD using the OpenEdition ISHELL File_systems pulldown menu, option 2.

The initial permissions of a newly created filesystem are 700, for security reasons. Therefore, the following steps are required only one time after creating the two filesystems:

```
use ISHELL to mount the filesystem on /vst
```

From a superuser shell, issue:

```
chmod 777 /vst/filesystem_name
```

Unmount the filesystem and repeat with the second.

The 700 cylinder HFS was mounted on /vst.

The following directories were created:

```
/vst/usr
/vst/usr/DTET2
/vst/usr/DTET2/vst
```

The following commands were issued from a superuser shell:

```
chown -R vst0:vstg0 /vst/usr
```

DTET2 2.3 was unwound in /vst/usr/DTET2 with the following command:

```
pax -o from=ISO8859-1,to=IBM-1047 -rzf /tmp/dtet23.pax.Z
```

The following changes to DTET2 were required:

```
diff -b /vst/usr/DTET2/src/dtet2/inetlib/accept.c /vst/usr/DTET2/vst/src/dtet2/inetlib/accept.c
34d33
<   $01 tedesco  08/22/96:  changed declare of len to type size_t
70,71c69
<      /* int len;                                                     */
<  size_t len;                  /* define as size_t          @01c */
---
>      int len;
```

```
diff -b /vst/usr/DTET2/src/dtet2/inetlib/connect.c /vst/usr/DTET2/vst/src/dtet2/inetlib/connect.c
48c48
<  $01=tedesco, 09/03/96:  added <arpa/inet.h> for ntohs
---
>
57d56
< #include <arpa/inet.h>                                  /* @01a */

diff -b /vst/usr/DTET2/src/dtet2/inetlib/tccdport.c /vst/usr/DTET2/vst/src/dtet2/inetlib/tccdport.c
34d33
<   $01=tedesco, 08/28/96:  added include of arpa/inet.h for ntohs
43d41
< #include <arpa/inet.h>                             /* @01a  */

diff -b /vst/usr/DTET2/src/dtet2/inetlib/tstcmenv.c /vst/usr/DTET2/vst/src/dtet2/inetlib/tstcmenv.c
35d34
<   $01=tedesco, 08/28/96:  added include of arpa/inet.h for ntohs
45d43
< #include <arpa/inet.h>                                 /* @01a */

diff -b /vst/usr/DTET2/src/dtet2/tcc/d_tcc_in.c /vst/usr/DTET2/vst/src/dtet2/tcc/d_tcc_in.c
40,41c40
<  $01 tedesco 8/22/96: changed declare of len from type 'int' to 'size_t'
<                       added <arpa/inet.h> for ntohs an ntohl
---
>
69d67
< #include <arpa/inet.h>                                  /* @01a */
140,142c138
<      /* int len, status;                                                */
<      int  status;                                               /* @01c */
<      size_t len;                                                /* @01a */
---
>      int len, status;
diff -b
diff -b /vst/usr/DTET2/src/dtet2/tccd/log.c /vst/usr/DTET2/vst/src/dtet2/tccd/log.c
34c34
<   $01=tedesco, 09/04/96:  move initialization of lfp out of declaration
---
>
57c57
< static FILE *lfp;                /* the log file stream pointer @01c */
---
> static FILE *lfp = stderr;              /* the log file stream pointer */
92d91
<         lfp = stderr;                                       /* @01a */

diff -b /vst/usr/DTET2/src/dtet2/xtilib/xtierror.c /vst/usr/DTET2/src/dtet2/xtilib/xtierror.c.orig
53c53
< {   /* changed sys_nerr to __sys_nerr     9/5/96    att */
---
> {
56,57c56,57
<      /* extern int sys_nerr;          not supported    att */
<      /* extern char *sys_errlist[];   not supported    att */
---
>      extern int sys_nerr;
>      extern char *sys_errlist[];
67,69c67,68
<             if (err > 0 &&; err < __sys_nerr)
<      /*            (void) fprintf(stderr, ": %s", sys_errlist[err]); */
<                    (void) fprintf(stderr, "attempted to print sys_errlist \n");
---
>             if (err > 0 &&; err < sys_nerr)
>                    (void) fprintf(stderr, ": %s", sys_errlist[err]);

diff -b /vst/usr/DTET2/src/dtet2/tcc/error.c /vst/usr/DTET2/src/dtet2/tcc/error.c.orig
58d57
< #include <errno.h>   /* required for __sys_nerr    9/5/96   att */
385d383
< /*  changed sys_nerr to __sys_nerr    9/5/96   att */
387,388c385,386
<      /* extern int sys_nerr;           not supported  att*/
<      /* extern char *sys_errlist[];    not supported  att */
---
>      extern int sys_nerr;
>      extern char *sys_errlist[];
396,398c394,396
<      if (errnum > 0 &&; errnum < __sys_nerr)
<             /* (void) sprintf(p, ": %s", sys_errlist[errnum]); */
<             (void) sprintf(p, "attempted to print sys_errlist \n");
```

```
---
>       if (errnum > 0 &&; errnum < sys_nerr)
>               (void) sprintf(p, ": %s", sys_errlist[errnum]);
>
diff -b /vst/usr/DTET2/src/dtet2/dtet2lib/prerror.c /vst/usr/DTET2/src/dtet2/dtet2lib/prerror.c.orig
50,52c50,51
< /* changed sys_nerr to __sys_nerr     9/5/96    att */
< /*    extern int sys_nerr;            not supported att*/
< /*    extern char *sys_errlist[];     not supported att */
---
>       extern int sys_nerr;
>       extern char *sys_errlist[];
57,59c56,57
<       if (errnum > 0 &&; errnum < __sys_nerr)
< /*            (void) fprintf(fp, ": %s", sys_errlist[errnum]); */
<               (void) fprintf(fp, "attempted to print sys_errlist\n");
---
>       if (errnum > 0 &&; errnum < sys_nerr)
>               (void) fprintf(fp, ": %s", sys_errlist[errnum]);

diff -b /vst/usr/DTET2/src/dtet2/inetlib/lhost.c /vst/usr/DTET2/src/dtet2/inetlib/lhost.c.orig
50c50
<       return(gethostaddr("localhos"));
---
>       return(gethostaddr("localhost"));
```

The archived vst4.1.4 test suites were loaded into the filesystems on the appropriate systems.

The test suite was unarchived in /vst/usr/DTET2/vst with the following command: On the master:

```
pax -o from=ISO8859-1,to=IBM-1047 -rzf /tmp/vst414.master.cpio.Z
```

On the slave:

```
pax -o from=ISO8859-1,to=IBM-1047 -rzf /tmp/vst414.slave.cpio.Z
```

Profile (**.profile**) file was created for vst0. The following environment variables were added to vst0's .profile:

```
export PATH=/bin:/usr/bin:/:$HOME/BIN:$HOME/../bin:$TET_ROOT/bin:.
export TET_ROOT=/vst/usr/DTET2
export TET_EXECUTE=$HOME/TESTROOT
```

./dtetcfg was executed, inet was selected as the transport type.

vst4.1.4 was then configured (**config.sh** was executed) on the master. The SRC/vsxparams file was verified.

The Makefiles were modified to meet the system requirements. The test suite was installed by issuing the following as vst0:

```
cd
unset _BPX_SHAREAS    /* required to use the su command   */
su root
make
```

SRC/userintf.ip was copied to **SRC/userintf.c** and was modified to meet our system requirements, on the master system.

The **defines.mk** file was modified to meet the system requirements, on the slave system.

usrintf.ip was copied to usrintf.c and was modified to meet our system requirements, on the slave system.

The tccd was started with the following command, issued as vst0, on the slave system:

```
tccd -u vst0
```

The test suite was built with the following command, issued as vst0, on the master system:

```
tcc -b
```

The test suite was executed.

```
tcc -e
```

Communicating Your Comments to IBM

OS/390
OpenEdition
XPG4 Conformance Document

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM. Whichever method you choose, make sure you send your name, address, and telephone number if you would like a reply.

Feel free to comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. However, the comments you send should pertain to only the information in this manual and the way in which the information is presented. To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

If you are mailing an RCF from a country other than the United States, you can give the RCF to the local IBM branch office or IBM representative for postage-paid mailing.

- If you prefer to send comments by mail, use the RCF at the back of this book.
- If you prefer to send comments by FAX, use this number:
  - FAX: (International Access Code)+1+914+432-9405
- If you prefer to send comments electronically, use this network ID:
  - IBMLink: (United States customers only): KGNVMC(MHVRCFS)
  - IBM Mail Exchange: USIB6TC9 at IBMMAIL
  - Internet e-mail: mhvrcfs@vnet.ibm.com
  - World Wide Web: http://www.s390.ibm.com/os390

Make sure to include the following in your note:

- Title and publication number of this book
- Page number or topic to which your comment applies

Optionally, if you include your telephone number, we will be able to respond to your comments by phone.

Reader's Comments — We'd Like to Hear from You

OS/390
OpenEdition
XPG4 Conformance Document

You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Your comments will be sent to the author's department for whatever review and action, if any, are deemed appropriate.

Note: Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.

Today's date: ______________________________________

What is your occupation?

Newsletter number of latest Technical Newsletter (if any) concerning this publication:

How did you use this publication?

[ ]	As an introduction	[ ]	As a text (student)
[ ]	As a reference manual	[ ]	As a text (instructor)
[ ]	For another purpose (explain)		

Is there anything you especially like or dislike about the organization, presentation, or writing in this manual? Helpful comments include general usefulness of the book; possible additions, deletions, and clarifications; specific errors and omissions.

Page Number: Comment:

Name	Address
Company or Organization	
Phone No.	

Reader's Comments — We'd Like to Hear from You

Cut or Fold
Along Line

Fold and Tape **Please do not staple** Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
Department 55JA, Mail Station P384
522 South Road
Poughkeepsie NY 12601-5400

Fold and Tape **Please do not staple** Fold and Tape

Cut or Fold
Along Line

Program Number: 5641-001

Printed in U.S.A.