UNIX Stadardization and Implementations

System Programming

2019 여름 계절학기

한양대학교 공과대학 컴퓨터소프트웨어학부
홍석준

❑ Proliferation of versions and differences leading to various standardization efforts

❑ Standardization

- ISO C
- IEEE POSIX
- The Single UNIX Specification
- FIPS

❑ Unix Implementations

- System V Release 4
- 4.4BSD
- FreeBSD
- Linux
- Mac OS X
- Solaris

I. ISO C

- **ANSI Standard X3.159-1989 for the C programming language**
  - ANSI: American National Standards Institute
  - ISO/IEC 9899:1990
    - International Organization for Standardization (ISO)

- **Portability of conforming C programs to a wide variety of operating systems, not just the Unix system**

- **The syntax and semantics of the programming language and a standard library**

- **ISO/IEC 9899:1999**

I. ISO C Standard

<assert.h>	verify program assertion
<complex.h>	complex arithmetic support
<ctype.h>	character types
<errno.h>	error codes
<fenv.h>	floating-point environment
<float.h>	floating-point constants
<inttypes.h>	integer type format conversion
<iso646.h>	alternate relational operator macros
<limits.h>	implementation constants
<locale.h>	locale categories
<math.h>	mathematical constants
<setjmp.h>	nonlocal goto
<signal.h>	signals
<stdarg.h>	variable argument lists
<stdbool.h>	boolean type and values
<stddef.h>	standard definitions
<stdint.h>	integer types
<stdio.h>	standard I/O library
<stdlib.h>	utility functions
<string.h>	string operations
<tgmath.h>	type-generic math macros
<time.h>	time and date
<wchar.h>	extended multibyte and wide character support
<wctype.h>	wide character classification and mapping support

- **POSIX (Portable Operating System Interface)**
- **"POSIX compliant" – not restricted to Unix and Unix-like systems**
- **No distinction bet'n sys calls and lib functions**
- **POSIX.1**
  - Originally IEEE Std 1003.1-1988
  - Revision published as IEEE Std 1003.1 – 1990 and ISO/IEC 9945-1:1990 → **POSIX.1**
- **IEEE Std 1003.1 – 2001**
  - Several 1003.1 amendments
  - 1003.2 standard
  - Portions of the Single UNIX Specification (SUS)
  - ISO/IEC 9899:1999, Programming Languages – C
- **Required & optional headers in Figure 2.2, 2.3, and 2.4**

I. POSIX.1

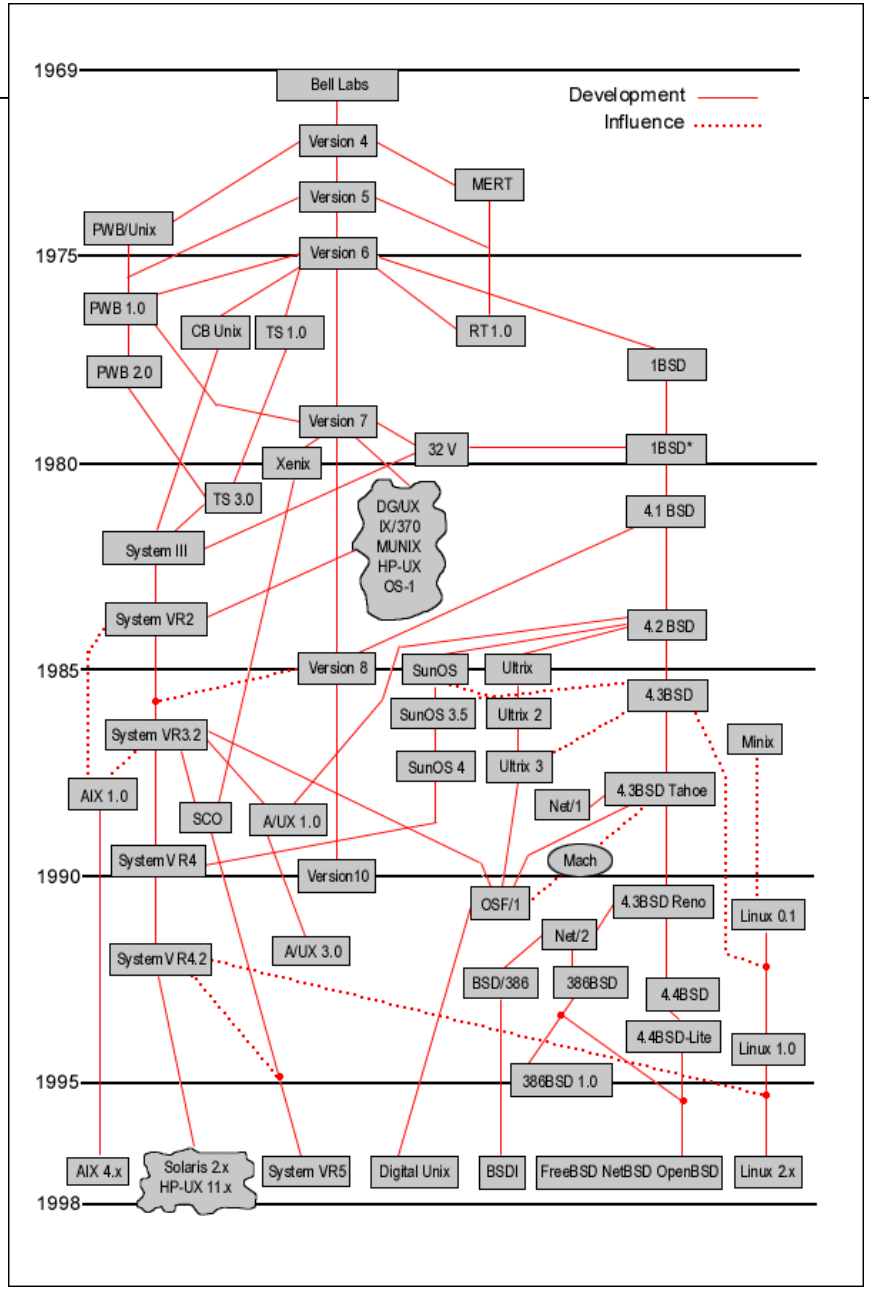
- <dirent.h> directory entries
- <fcntl.h> file control
- <fnmatch.h> filename-matching types
- <glob.h> pathname pattern-matching types
- <grp.h> group file
- <netdb.h> network database operations
- <pwd.h> password file
- <regex.h> regular expressions
- <tar.h> tar archive values
- <termios.h> terminal I/O
- <unistd.h> symbolic constants
- <utime.h> file times
- <wordexp.h> word-expansion types
- <arpa/inet.h> Internet definitions
- <net/if.h> socket local interfaces
- <netinet/in.h> Internet address family
- <netinet/tcp.h> Transmission Control Protocol definitions
- <sys/mman.h> memory management declarations
- <sys/select.h> select function
- <sys/socket.h> sockets interface
- <sys/stat.h> file status
- <sys/times.h> process time
- <sys/types.h> primitive system data types
- <sys/un.h> UNIX domain socket definitions
- <sys/utsname.h> system name
- <sys/wait.h> process control

❑ X/Open System Interface (XSI)

- A superset of the POSIX.1 standard
- Only XSI-conforming implementations can be called UNIX systems.
- SUSv1 (a.k.a. Spec 1170) in 1994
- SUSv2 in 1997 added support for threads, real-time interfaces, 64-bit processing, large files, and enhanced multibyte character processing.
- SUSv3 in 2001
  - Base Specifications (the same as the IEEE Std 1003.1-2001)
    - Base Definitions, System Interfaces, Shell and Utilities, and Rationale
  - X/Open Curses Issue 4, Version 2
- SUSv3 2004 Edition

I. Unix Family Tree

- http://www.isk.kth.se/kursinfo/6b2019/for/f1/unixhistory.gif



I. System V Release 4

- **1989 by AT&T's Unix System Laboratories**
- **A merging of SVR3.2, SunOS, 4.3BSD, and Xenix**
- **POSIX 1003.1 and X/OPEN XPG3 compliant**
- **SVID (System V Interface Definition)**
  - Issue 3 defines functionality to qualify as SVR4

I. 4.4BSD

- Berkeley Software Distribution
- 4.2BSD in 1983, 4.3BSD in 1986, 4.3BSD Tahoe in 1988, 4.3Reno in 1990, and 4.4BSD in 1992
- BSD Networking Software Release 1.0 in 1989 (from BSD Tahoe) and Release 2.0 in 1991 (from BSD Reno)
- 4.4BSD-Lite Release 1 in 1994
- 4.4BSD-Lite Release 2 in 1995

I. Other Implementations

- **FreeBSD based on 4.4BSD-Lite**
- **Linux in 1991 as a grass-root effort**
- **Mac OS X**
  - The core OS is called *Darwin*, which is a FreeBSD/Mach hybrid.
- **Solaris based on SVR4**
- **AIX (IBM), HP-UX (HP), IRIX (Silicon Graphics), etc.**

- **To aid the portability of software**
- **Compile-time options and limits (headers)**
  - Does the system support job control?
  - What's the largest value of a short integer?
- **Run-time limits not associated with a file or directory (sysconf function)**
- **Run-time limits associated with a file or directory (pathconf and fpathconf function)**
  - How many characters in a filename?

I. <limits.h>

❑ ISO C Limits (Figure 2.6)

- …
- LONG_MAX max value of long 2,147,483,647
- LONG_MIN min value of long -2,147,483,647
- ULONG_MAX max value of unsigned long 4,294,967,295
- …

❑ POSIX limits (Figure 2.8)

- …
- _POSIX_OPEN_MAX number of open files per process 20
- _POSIX_PATH_MAX number of bytes in a pathname 256
- …

❑ XSI limits (Figure 2.9)

- …
- NL_ARGMAX max value of digit in calls to printf/scanf 9
- NL_LANGMAX max no. of bytes in LANG environ variable 14
- …

#include <unistd.h>
long sysconf(int *name*);
long pathconf(const char **pathname*, int *name*);
long fpathconf(int *filedes*, int *name*);

- **The name argument (Figure 2.10 & 2.11)**
  - …
  - _SC_ARG_MAX max length of arguments to `exec()`
  - _SC_CHILD_MAX max no. of processes per real user ID
  - _SC_CLK_TCK no. of clock ticks per second
  - _SC_OPEN_MAX max no. of open files per process
  - …
  - _PC_NAME_MAX max no. of bytes in a filename
  - _PC_PATH_MAX max no. of bytes in a relative pathname
  - …

- **Feature test macros – no any implementation-defined limits**

- **_POSIX_C_SOURCE and _XOPEN_SOURCE**
  - cc –D_POSIX_C_SOURCE=200112 file.c
  - #define _POSIX_C_SOURCE 200112

- **__STDC__ - automatically defined by an ISO C compiler**

```
#ifdef __STDC__
void *myfunc(const char *, int);
#else
void *myfunc();
#endif
```

❑ <sys/types.h> defines some implementation-dependent data types.

- caddr_t core address
- clock_t counter of clock ticks
- comp_t compressed clock ticks
- dev_t device numbers (major and minor)
- fd_set file descriptors sets
- fpos_t file position
- gid_t numeric group IDs
- ino_t i-node numbers
- …

❑ Major and minor device numbers

- Historically, 8 bits for both major and minor device no
- Solaris: 14 bits for the major and 18 bits for the minor

Q and A