Towards a UNIX Standard

Michael Tilson
Human Computing Resources Corporation
10 St. Mary Street
Toronto, Canada, M4Y 1P9
416-922-1937
{decvax,utzoo,utcsrgv}!hcr!hcrvax!mike

The great flexibility of the UNIX system together with the peculiar history of AT&T releases have combined to produce a number of somewhat incompatible commercial and academic versions of UNIX. From the chaotic multitude of UNIX versions a demand for standardization has arisen. This demand also coincides with current AT&T product marketing directions.

However, the impetus towards standardization has created a sense of unease in the original UNIX community. Support and marketing groups at AT&T now seem to control the future direction, rather than the perhaps more tasteful research groups. There is a fear that innovation will be stifled. As a result, standardization proposals are met with some resistance, particularly standardization which may appear to be based upon well financed marketing efforts rather than technical excellence.

Despite the dangers, there are good reasons to believe that standards will prepare the groundwork for the next wave of innovation, and that benefits of a standard far exceed the costs.

UNIX History

Since UNIX is the product of a single vendor (AT&T), one might think that a standard should not be necessary. However, the history of the system has caused a large degree of divergence. It is important to review the features of this history in order to understand the background of current standardization efforts. The following history is somewhat simplified, but adequate for the purpose. (Those familiar with the history of UNIX versions may skip this section.)

The UNIX system was developed by Ken Thompson and Dennis Ritchie of Bell Laboratories as a personal tool for program development using only the most limited hardware environment. The system was a synthesis of many existing good ideas, and a few new ones. Because the basic conception was the product of only two minds, the early UNIX system was especially elegant and exhibited a unity of design rarely found in larger-scale systems.

This system was used internally at Bell Labs, and was gradually improved. It came to the attention of the academic community, and was released at no charge and totally without support for educational purposes. The first release that had any significant distribution was the "Fifth Edition", now known as "Version 5". (At that time, there were no UNIX "versions". Instead, the Programmer's Manual was updated with a new edition as the system changed. With each new edition of the manual, a tape copy of the research computer system disk was made. This essentially became the distribution tape.)

By the time of the first outside release, the system had been coded in the "C" programming language, but it ran only on the DEC PDP-11 processor.

The "Sixth Edition" ("Version 6") had a fuller set of functions. With this version, AT&T allowed commercial users to purchase source copies of the system. The price was fairly high ($20,000), and the system came "as-is", with absolutely no support. It still ran on the PDP-11, and only a limited number of configurations were supported. Version 6 spread rapidly in some academic circles, and began to be used in commercial and government projects.

Version 7 of UNIX provided a fuller set of features. In addition, much of the system had been re-designed to allow portability. Internally within Bell Labs the system had been re-targeted ("ported") to another processor type. AT&T started to realize that the system had commercial potential, and allowed for binary redistribution of the system by independent vendors at a lower price. Still, no customer support was provided.

UNIX 32V was the only "port" of Version 7 to be released. It supported the 32-bit DEC Vax architecture. Otherwise, it was nearly unchanged Version 7. Only a very limited number of Vax configurations were supported.

Version 7 was the last version of UNIX ever to be released to the outside world by the original Bell Labs "research" group. There is a Version 8 internally, but it is unobtainable. Thompson and Ritchie are still associated at times with UNIX system development, but only within this research area. Version 7 was the basis of all of the early "commercial" versions of UNIX. The system was ported to various microprocessor types by a number of companies, and offered with commercial support in binary form.

After Version 6, the first of many splits in the line of UNIX development occurred. Internal Bell System users wanted UNIX, but needed a more solid "commercial" product. For this reason, Western Electric produced the "Programmer's Workbench" or "PWB" system. This ran on the PDP-11, and included a number of programmer productivity tools, as well as much more complete support for a variety of PDP-11 hardware. PWB was essentially an internal Bell System supported product. It was available to the external world, again with no support. The PWB line of development is the basis of the "USG" ("UNIX Support Group") systems, and may perhaps be considered the direct

ancestor of all other UNIX systems available from AT&T.

All of these releases were made years after each version had been made available internally to the Bell System.

UNIX spread rapidly in the academic world. The system was intellectually elegant, easy to use for research purposes, free, simple, and available in source code form so that changes needed for computer science research purposes could be accommodated. Most universities had UNIX machines, which were typically run by unpaid student "UNIX gurus". The system was easy to change, had no standards, and had no support contracts which might become invalid. The result was a torrent of changes and features, some useful, many not. Early meetings of what was to become Usenix concentrated on these changes, and many tapes were traded back and forth.

When UNIX 32V was released, a number of academic institutions were upgrading from 16 to 32 bit computers. 32V was a very rough release. It did not support many Vax configurations. It was difficult to install. It was oriented towards older hardware (since the release was two years old). Certain productivity features were missing, such as screen editing. The demand paging ability of the Vax was not used. The University of California at Berkeley ("UCB") set out to rectify these problems. The result was a series of UNIX releases from UCB, called "Berkeley Software Distributions". Release numbers are in the form "4.xBSD". The UCB effort resulted in increased performance, greater hardware support, demand paging, useful features, and a myriad of additions, new options, experimental ideas, and simply gratuitous changes. UCB included many of the features that had been added elsewhere, and seemed to make an effort to provide as many programs as possible on its distribution tapes, which were made available for a handling charge to any UNIX source licensee. The most widely used version was 4.1BSD. Most Vax UNIX sites used this system in preference to 32V, since it had higher performance, and was likely to boot on a typical Vax configuration without any custom modifications. 32V on the other hand could only boot on a specific limited set of configurations.

UCB had obtained funding to meet the UNIX needs of Darpa (Defense Advanced Research Projects Agency). These needs included high throughput image processing, and support of Darpa standards for local and long haul networks. At the same time, UCB set out to "fix" many things that were "wrong" or "missing" in UNIX. The culmination of this effort is the 4.2BSD system.

In the meantime, AT&T had been busy. UNIX System III provided lower binary pricing, and most of the features of PWB and Version 7 combined. (Although released after Version 7, it was part of the USG line of descent, and it in fact did not include some V7 features.) Both the PDP-11 and Vax were supported. This was touted as the first "commercial" UNIX offering. However, the distribution was hurriedly packaged, and the documentation reached a new low of inconsistency. For the first time, AT&T started to actively market the system. AT&T marketing created a demand for "System III". Most commercial vendors

added the important additional System III functions, but few gave up their already ported and reliable Version 7 bases. Commercial systems became Version 7/System III hybrids. Most commercial vendors also included Berkeley functions. All vendors sold software under System III licensing provisions in order to obtain the lowest price, even if the actual system sold was based on an earlier UNIX version. (System III licensing included a license for all previous versions as a subset.)

System V followed System III. With System V, AT&T brought the outside world up to date with the internal USG version. (System IV was skipped.) The system was cleaned up, and the documentation put into better order. Certain Berkeley features were added, and performance issues were addressed. Full scale marketing commenced. Full commercial support was provided to source customers. New features were added.

System V was largely upward compatible with System III. Previously, successive releases had been seriously incompatible. System V Release 2 is now available. It represents a truly upward compatible addition to System V.

Today, there are a variety of commercial UNIX systems. Most commercial vendors are moving towards System V. However, successive releases, hybrids, upgrades, offspring, proprietary enhancements, and vendor subsetting have resulted in a situation today in which end-users are unsure about what they are purchasing when they buy "UNIX".

In the academic world, many sites are moving to 4.2BSD at the moment. This is a natural outgrowth of the fact that Vaxes are the academic computer of choice, and BSD systems were for a long time the only available option to support a variety of Vax configurations. However, the BSD stream has become quite divergent from the USG stream, creating many conversion and compatibility problems. It is now misleading to the uninitiated to have both systems called "UNIX". This has been a source of a lot of confusion.

Despite the variations, it is important to remember that all of the systems, even 4.2BSD, remain recognizably derived from "UNIX".

Emerging UNIX Standards

The UNIX system has many advantages which explain its popularity: It is multi-process, multi-user, powerful, elegant, and portable. Here we will focus on portability and standardization.

The UNIX system is highly (although not perfectly) portable. UNIX implementations exist (or are in progress) on a multitude of machines with varying architecture. A related feature is the system's span of usability. A UNIX programmer can use anything from a cheap desktop micro to a giant supercomputer. To a large extent, the procedures used by the UNIX programmer will be exactly identical on each machine. There is no other available system which can accommodate a factor of

10,000 difference in machine power. The same applications programs can be used.

As we have seen, this portability is marred inconsistency and lack of standards. Commercial UNIX systems vary unnecessarily from one vendor or machine to another. There are few standards for command options, error messages, etc. It should also be noted that there is no binary compatibility. Applications software vendors can not produce one UNIX binary for each CPU type (e.g. Motorola 68000). This fragments and inhibits the applications software market, to the detriment of vendors, and more importantly, to the detriment of users wishing to purchase to best available, lowest cost software.

UNIX should be viewed as a vehicle for applications programs. UNIX systems provide an applications software architecture which is independent of machine details, including processor instruction sets, memory management schemes, and I/O architecture. UNIX is a "software bus", which allows applications to move without change. It should not be necessary for the typical application to use any system function which is variable from system to system. In particular, this means there must be a minimum set of functions which are guaranteed to exist on all implementations, and guaranteed to have the same specifications on all implementations. If these criteria are satisfied, application software can be developed and tested once. From that point on, it can be moved from environment to environment without further attention.

The unnecessary variability of UNIX systems hinders the growth of UNIX usage. The system buyer would like to know that it is enough to specify "UNIX" in a purchase order, without having to ask questions like "Does it have sh2?" or "Are FIFO files implemented?". These questions are irrelevant to the end user of applications software, even though the answers are highly relevant to whether the software can actually run.

A software standard is a treaty between producers and users of a system. From the producer or supplier end, a standard specifies the functions which must be present. Other functions can be supplied, but the standard specifies those that must always be supplied, and how these should work. From the user end, the standard specifies the only functions that can be assumed if software is to be portable to all standard conforming systems. In this paper the "producer" is the supplier of the systems software. The "user" is the writer of application software, either a software vendor or an end user. From the "applications software bus" point of view, a standard defines an exact meeting point; a minimum for suppliers and a maximum for users.

Although this is the most important area for initial standardization, there can be many levels of "producers" and "users" involved. For example, the authors of the systems software are in turn consumers or users of device hardware. Each level of producer/user relationship can be the topic of a standard. It is advantageous to prevent intermingling of the levels in one standards document, unless each level is clearly defined.

Standards promote stability and reusability of software. A system standard decouples decisions made by applications software producers from decisions made by the system producer. The standard allows each to conduct business in an orderly way, with a minimum of conflict. The purpose of a software standard is inherently related to portability, either from machine to machine or in time -- from version to version. The portability issue is what distinguishes a standard from a mere product specification.

Standards are introduced for economic reasons. One can always rewrite software for different systems, but it is much more economic to avoid this. Once a standard becomes widely accepted, the economic impact of incompatible change becomes so large that change is almost unthinkable. As a result, there is inherent conflict between innovation and standardization.

Due to the unchanging nature of a standard and the large potential economic consequences, the most important standardization decisions have to do with what is left out, rather than with what is included. Once something is in, it is in forever. Some inclusions are less dangerous than others. A specification that a particular UNIX command is to be included is not terribly dangerous. If a better command comes along, the old one can still be supplied. User manuals can indicate the commands which are desirable, as opposed to those which are included solely for standard conformance. On the other hand, some standardized items rule out other solutions. For example, if the syntax of file names is completely specified, other naming conventions are forbidden. Here great care must be exercised.

Despite its variability, the UNIX system is already widely viewed as a standard. There is general industry demand for standardization. This demand is being encouraged by AT&T. The /usr/group Standard represents one result of industry demand. Other members of the industry are promoting UNIX software standardization. A recent example is the "ISIS" standardization effort proposed by several applications software vendors.

The new "commercial" AT&T recognizes the need to regain control of its UNIX product. Therefore, AT&T is heavily promoting sales and support of its latest UNIX release, System V. There is a commitment to upward-compatibility in future releases. AT&T has arranged that all four major microcomputer chip makers will support a standard certified version of System V. Few will have missed the massive advertising campaign, carrying the message in publications ranging from the Wall Street Journal and Scientific American to the computer trade press. The message is: System V is the one true UNIX, and the only version worth having. Of course, this message is not technically true, but the marketing campaign is having a strong effect, especially since it fits directly with customer desires for standardization.

In the area of UNIX standards, the following appear likely:

- UNIX System V will form the basis of the standards.

- o AT&T will back standardization, and will maintain its commitment to upward compatibility.
- o The /usr/group Standard will be formally adopted in 1984.
- o Vendors and purchasers in the business market will demand standard-conforming systems. In particular, government purchasing requirements will specify standard systems, initially by specifying AT&T standards, and later by using the government's own standards or formal industry standards.
- o ANSI and international standards organizations will, in the longer term, adopt a standard. (Note: ANSI X3J11 is working on the C language standard, which will include many standard UNIX library routines.)
- o Academic usage will converge towards standardization over time as conforming systems become more functional.
- o UNIX will become truly generic -- the industry standard interface between applications software and computer systems. (Note: as a result, AT&T may eventually have trouble protecting both the UNIX "trade secret" and the UNIX trademark. Effective standards will also deter arbitrary change by AT&T itself, for example, changes intended only to promote sale of "3B" hardware.)

The /usr/group Standard

The /usr/group Standards Committee has been active since 1981. The committee has wide representation from the UNIX industry. Committee members serve as individuals, but the voting members include representatives from approximately 40 organizations, including AT&T.

The /usr/group Standard is a semi-formal document. It has not gone through the treatment given by an international standards body such as ISO. Nevertheless, the Committee has given a great deal of consideration to a number of issues. The purpose of the Standard was to provide the set of features needed by the vast majority of applications software, at the systems interface (i.e. C subroutine call) level. Although stated to be derived from System III, the current document is essentially also a subset of System V, with greater precision, and with one extension. The Standard was not intended as a "UNIX Standard", but rather as an operating systems interface standard derived from UNIX and UNIX-like systems. (In practice, however, it is a UNIX standard.)

The final draft standard was approved by an overwhelming roll-call vote at a Committee meeting held concurrent with the January 1984 UniForum conference in Washington, D.C. The AT&T representatives voted for approval of the Standard. At this writing, the document is before the voting members of /usr/group for final approval.

The main areas of debate were:

- Size. Some members wished to have a more minimal set, while others wished to include as much of UNIX as possible.

- Change. Committee members often wished to repair perceived problems in UNIX, or to make the system "better". To a large extent, this was resisted.

- Precision vs. speed. The Committee felt that it had to complete its work in a timely fashion. The Standard is more informal and less precise than some other standard documents. This decision probably cut two years off the publication time.

- Conformance with current practice, backward compatibility. Some redundant items are included in the standard (e.g. both "dup" and "fcntl" system calls), in order to preserve the large body of existing applications programs.

- Implementation dependencies. The committee tried to define the standard so that it could be implemented without standardizing undocumented idiosyncrasies of the Bell Labs code. UNIX "look-alike" vendors lobbied for changes to accommodate various "improvements" which existed in those vendor products. Other vendors lobbied against deviation from AT&T.

- Extensions. Various extensions to the system have been proposed. Only the "record locking" extension was accepted. (Note: we are informed that a future release of UNIX from AT&T will contain this feature as an interface to a somewhat larger facility.)

- UCB compatibility. Two members of the Committee argued against final adoption of the Standard on the grounds that it may be difficult to support under 4.2BSD, and in particular that standard features such as FIFO files created problems in networking environments. Most members of the Committee felt that this latter issue represented a deficiency in some particular networking code, rather than a fundamental objection.

The AT&T representatives have informally indicated that AT&T intends to conform to the standard in future releases.

The Standard has some trouble spots, and work remains:

- Terminal control is not standardized. Many vendors had different conventions derived from different UNIX versions. It is expected that a future update to the Standard will contain a System V subset, chosen so as to be reasonably easy to implement on most existing UNIX implementations.

- Commands are not standardized. Only the "systems interface" is now defined. This makes it difficult for applications software to take advantage of the UNIX "tool building" philosophy by calling existing commands.

- The Standard is at the source level. This is the place to start, but lack of binary compatibility is a current drawback in the UNIX market.

- There likely remain some internal inconsistencies and fuzzy areas. These are minor, but annoying.

Dangers and Opportunities

Despite any problems, the proposed Standard represents wide industry agreement, and is a very hopeful sign in the development of the UNIX market. The availability of a standardized operating environment will dramatically heighten competition in the computer industry. As long as the chosen processor runs a compatible version of UNIX, a user can feel free to mix and match the hardware for best performance. At the low end, there are now approaching a hundred 32-bit UNIX micros to choose from. The high end choices are more limited, but a number of new companies are going after the "three times a 780" market, and the larger companies are expected to introduce machines in that range. Users will finally be free of vendor "lock-in" via proprietary systems. Price and performance will be the main issues. Compatibility means freedom.

The impetus towards standardization has created a sense of unease in the original UNIX community. UNIX development has always been characterized by a spirit of innovation, and there is a feeling that innovation will be now be stifled. As a result, standardization proposals are met with some resistance. Despite the dangers, there are good reasons to believe that standards will prepare the groundwork for the next wave of innovation.

Systems evolve from an initial pure concept, through the first commercial realization, into maturity, and finally into senility. A good initial concept is usually one that synthesizes and unifies many previously disparate features, but which does not attempt to solve all of the world's problems. (For example, the one UNIX ordinary file type can conceptually replace most file types found on other systems in most applications.) As a system matures, additional features are added to fill out the functional range. However, as Rob Pike notes, the system gets five times bigger without getting five times better. Eventually, no amount of addition, hacking, or patching can be superior to the next pure synthesis. It is most useful to standardize before this point is reached, and get on with the research necessary to generate the next concept, rather than hastening the senility of the current one.

A standard operating environment will in fact provide a solid reliable basis upon which software developers can build. UNIX represents a much better standard than most. It is time to stop solving problems by changing the kernel calls or changing the meaning of options. One must recognize that, although a change may make the system better, it has to be enough better to be worth being different.

Finally, it is good to keep in mind that a "UNIX standard" does not prevent wholesale change or innovation. The UNIX system will continue to provide be a fertile ground for experimentation. A standard may prevent calling the result "UNIX", but that in itself is not a bad thing.

What Next?

The computer industry is at an historic turning point. If industry agreement on systems standards is achieved, users will be able to freely choose among a variety of hardware architectures. Programmers will no longer waste time adapting software to new environments, but instead will have more time to engage in creative work. The software industry will have a hitherto unparalleled degree of software portability.

The actions of AT&T and IBM will have a big effect on the emerging industry standards. AT&T has already released its new computer products, all of which run UNIX System V. But the biggest force is of course IBM. IBM has experienced tremendous success with a "generic" operating system on the PC. IBM is moving forward on the UNIX front, with announcements for PC/IX for the PC and Xenix for the 9002. By the time this article goes to press, there will likely be more announcements.

Will IBM conform to the AT&T standard, will it go with 4.2BSD, will it go with some other radical variant, or will it go with its own proprietary portable system? Unlike many other companies, IBM is large enough to be able to do all of this at once. Since each IBM division operates with some degree of autonomy, we should expect in the short term a little bit of everything. However, keep in mind the fact that IBM can well afford any conceivable proprietary operating systems development. In the long run, the only reason for IBM to go with UNIX is because the market demands generic systems. Since UNIX is so portable, if IBM does not supply it, somebody else will. This will increase the incentive for IBM to jump on the bandwagon.

On the other hand, if wide industry agreement on UNIX standards does not solidify quickly, IBM will perhaps be free to push forward its own proprietary software as some sort of industry standard. This possibility represents a grave threat to computer users, far worse than any inconvenience caused by minor technical flaws in a standardized version of UNIX.