The Strange Birth and Long Life of Unix

The classic operating system turns 40, and its progeny abound

By WARREN TOOMEY / DECEMBER 2011

Photo: Alcatel-Lucent

KEY FIGURES: Ken Thompson [seated] types as Dennis Ritchie looks on in 1972, shortly after they and their Bell Labs colleagues invented Unix.

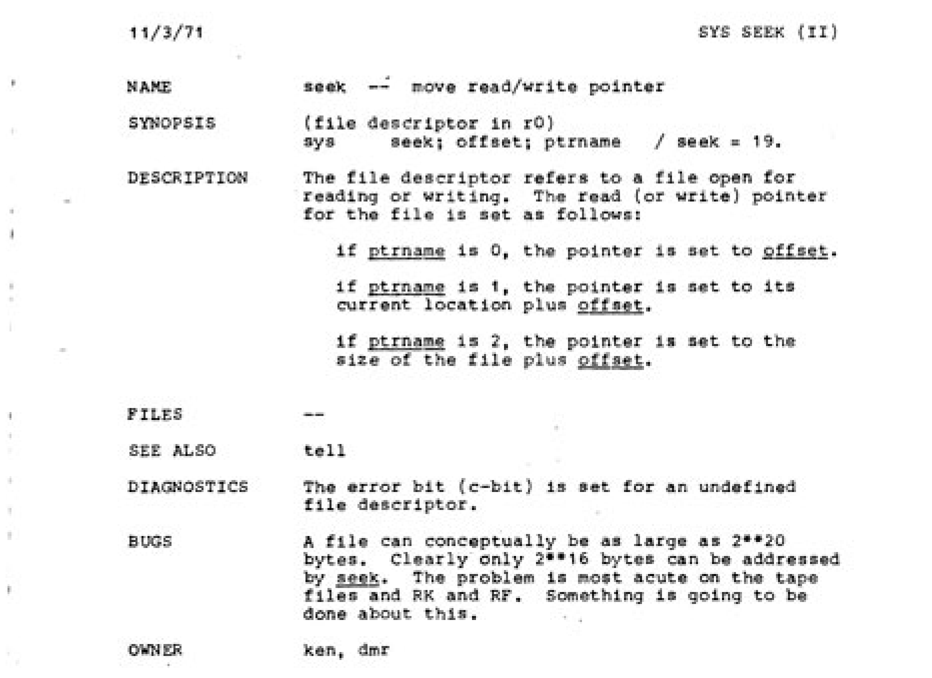
They say that when one door closes on you, another opens. People generally offer this bit of wisdom just to lend some solace after a misfortune. But sometimes it's actually true. It certainly was for Ken Thompson and the late Dennis Ritchie, two of the greats of 20th-century information technology, when they created the Unix operating system, now considered one of the most inspiring and influential pieces of software ever written.

A door had slammed shut for Thompson and Ritchie in March of 1969, when their employer, the American Telephone & Telegraph Co., withdrew from a collaborative project with the Massachusetts Institute of Technology and General Electric to create an interactive time-sharing system called Multics, which stood for "Multiplexed Information and Computing Service." Time-sharing, a technique that lets multiple people use a single computer simultaneously, had been invented only a decade earlier. Multics was to combine time-sharing with other technological advances of the era, allowing users to phone a computer from remote terminals and then read e-mail, edit documents, run calculations, and so forth. It was to be a great leap forward from the way computers were mostly being used, with people tediously preparing and submitting batch jobs on punch cards to be run one by one.

Over five years, AT&T invested millions in the Multics project, purchasing a GE-645 mainframe computer and dedicating to the effort many of the top researchers at the company's renowned Bell Telephone Laboratories—including Thompson and Ritchie, Joseph F. Ossanna, Stuart Feldman, M. Douglas McIlroy, and the late Robert Morris. But the new system was too ambitious, and it fell troublingly behind schedule. In the end, AT&T's corporate leaders decided to pull the plug.

After AT&T's departure from the Multics project, managers at Bell Labs, in Murray Hill, N.J., became reluctant to allow any further work on computer operating systems, leaving some researchers there very frustrated. Although Multics hadn't met many of its objectives, it had, as Ritchie later recalled, provided them with a "convenient interactive computing service, a good environment in which to do programming, [and] a system around which a fellowship could form." Suddenly, it was gone.

With heavy hearts, the researchers returned to using their old batch system. At such an inauspicious moment, with management dead set against the idea, it surely would have seemed foolhardy to continue designing computer operating systems. But that's exactly what Thompson, Ritchie, and many of their Bell Labs colleagues did. Now, some 40 years later, we should be thankful that these programmers ignored their bosses and continued their labor of love, which gave the world Unix, one of the greatest computer operating systems of all time.



```
11/3/71                                              SYS SEEK (II)

NAME            seek  --  move read/write pointer

SYNOPSIS        (file descriptor in r0)
                sys     seek; offset; ptrname   / seek = 19.

DESCRIPTION     The file descriptor refers to a file open for
                reading or writing.  The read (or write) pointer
                for the file is set as follows:

                   if ptrname is 0, the pointer is set to offset.

                   if ptrname is 1, the pointer is set to its
                   current location plus offset.

                   if ptrname is 2, the pointer is set to the
                   size of the file plus offset.

FILES           --

SEE ALSO        tell

DIAGNOSTICS     The error bit (c-bit) is set for an undefined
                file descriptor.

BUGS            A file can conceptually be as large as 2**20
                bytes.  Clearly only 2**16 bytes can be addressed
                by seek.  The problem is most acute on the tape
                files and RK and RF.  Something is going to be
                done about this.

OWNER           ken, dmr
```

MAN MEN: Thompson (ken) and Ritchie (dmr) authored the first Unix manual or "man" pages, one of which is shown here. The first edition of the manual was released in November 1971. *Click to enlarge.*

The rogue project began in earnest when Thompson, Ritchie, and a third Bell Labs colleague, Rudd Canaday, began to sketch out on paper the design for a file system. Thompson then wrote the basics of a new operating system for the lab's GE-645 mainframe. But with the Multics project ended, so too was the need for the GE-645. Thompson realized that any further programming he did on it was likely to go nowhere, so he dropped the effort.

Thompson had passed some of his time after the demise of Multics writing a computer game called *Space Travel*, which simulated all the major bodies in the solar system along with a spaceship that could fly around them. Written for the GE-645, Space Travel was clunky to play—and expensive: roughly US $75 a game for the CPU time. Hunting around, Thompson came across a dusty PDP-7, a minicomputer built by Digital Equipment Corp. that some of his Bell Labs colleagues had purchased earlier for a circuit-analysis project. Thompson rewrote *Space Travel* to run on it.

And with that little programming exercise, a second door cracked ajar. It was to swing wide open during the summer of 1969 when Thompson's wife, Bonnie, spent a month visiting his parents to show off their newborn son. Thompson took advantage of his temporary bachelor existence to write a good chunk of what would become the Unix operating system for the discarded PDP−7. The name Unix stems from a joke one of Thompson's colleagues made: Because the new operating system supported only one user (Thompson), he saw it as an emasculated version of Multics and dubbed it "Un-multiplexed Information and Computing Service," or Unics. The name later morphed into Unix.

Initially, Thompson used the GE-645 to compose and compile the software, which he then downloaded to the PDP−7. But he soon weaned himself from the mainframe, and by the end of 1969 he was able to write operating-system code on the PDP-7 itself. That was a step in the right direction. But Thompson and the others helping him knew that the PDP−7, which was already obsolete, would not be able to sustain their skunkworks for long. They also knew that the lab's management wasn't about to allow any more research on operating systems.

So Thompson and Ritchie got creative. They formulated a proposal to their bosses to buy one of DEC's newer minicomputers, a PDP-11, but couched the request in especially palatable terms. They said they were aiming to create tools for editing and formatting text, what you might call a word-processing system today. The fact that they would also have to write an *operating* system for the new machine to support the editor and text formatter was almost a footnote.

Management took the bait, and an order for a PDP-11 was placed in May 1970. The machine itself arrived soon after, although the disk drives for it

Photo: Gabriela Hasbun

PROUD FATHER: Ken Thompson poses to show off his brainchild, 40 years after its birth.

took more than six months to appear. During the interim, Thompson, Ritchie, and others continued to develop Unix on the PDP-7. After the PDP-11's disks were installed, the researchers moved their increasingly complex operating system over to the new machine. Next they brought over the roff text formatter written by Ossanna and derived from the runoff program, which had been used in an earlier time-sharing system.

Unix was put to its first real-world test within Bell Labs when three typists from AT&T's patents department began using it to write, edit, and format patent applications. It was a hit. The patent department adopted the system wholeheartedly, which gave the researchers enough credibility to convince management to purchase another machine—a newer and more powerful PDP-11 model—allowing their stealth work on Unix to continue.

During its earliest days, Unix evolved constantly, so the idea of issuing named versions or releases seemed inappropriate. But the researchers did issue new editions of the programmer's manual periodically, and the early Unix systems were named after each such edition. The first edition of the manual was completed in November 1971.

So what did the first edition of Unix offer that made it so great? For one thing, the system provided a hierarchical file system, which allowed something we all now take for granted: Files could be placed in directories—or equivalently, folders—that in turn could be put within other directories. Each file could contain no more than 64 kilobytes, and its name could be no more than six characters long. These restrictions seem awkwardly limiting now, but at the time they appeared perfectly adequate.

Although Unix was ostensibly created for word processing, the only editor available in 1971 was the line-oriented ed. Today, ed is still the only editor guaranteed to be present on all Unix systems. Apart from the text-processing and general system applications, the first edition of Unix included games such as blackjack, chess, and tic-tac-toe. For the system administrator, there were tools to dump and restore disk images to magnetic tape, to read and write paper tapes, and to create, check, mount, and unmount removable disk packs.

Most important, the system offered an interactive environment that by this time allowed time-sharing, so several people could use a single machine at once. Various programming languages were available to them, including BASIC, Fortran, the scripting of Unix commands, assembly language, and B. The last of these, a descendant of a BCPL (Basic Combined Programming Language), ultimately evolved into the immensely popular C language, which Ritchie created while also working on Unix.

Photo: Mark Richards/Computer History Museum

The first edition of Unix let programmers call 34 different low-level routines built into the operating system. It's a testament to the system's enduring nature that nearly all of these system calls are still available—and still heavily used—on modern Unix and Linux systems four decades on. For its time, first-edition Unix provided a remarkably powerful environment for software development. Yet it contained just 4200 lines of code at its heart and occupied a measly 16 KB of main memory when it ran.

Unix's great influence can be traced in part to its elegant design, simplicity, portability, and serendipitous timing. But perhaps even more important was the devoted user community that soon grew up around it. And that came about only by an accident of its unique history.

The story goes like this: For years Unix remained nothing more than a Bell Labs research project, but by 1973 its authors felt the system was mature enough for them to present a paper on its design and implementation at a symposium of the Association for Computing Machinery. That paper was published in 1974 in the *Communications of the ACM*. Its appearance brought a flurry of requests for copies of the software.

This put AT&T in a bind. In 1956, AT&T had agreed to a U.S government consent decree that prevented the company from selling products not directly related to telephones and telecommunications, in return for its legal monopoly status in running the country's long-distance phone service. So Unix could not be sold as a product. Instead, AT&T released the Unix source code under license to anyone who asked, charging only a nominal fee. The critical wrinkle here was that the consent decree prevented AT&T from supporting Unix. Indeed, for many years Bell Labs researchers proudly displayed their Unix policy at conferences with a slide that read, "No advertising, no support, no bug fixes, payment in advance."

With no other channels of support available to them, early Unix adopters banded together for mutual assistance, forming a loose network of user groups all over the world. They had the source code, which helped. And they didn't view Unix as a standard software product, because nobody seemed to be looking after it. So these early Unix users themselves set about fixing bugs, writing new tools, and generally improving the system as they saw fit.

The Usenix user group acted as a clearinghouse for the exchange of Unix software in the United States. People could send in magnetic tapes with new software or fixes to the system and get back tapes with the software and fixes that Usenix had received from others. In Australia, the University of Sydney produced a more robust version of Unix, the Australian Unix Share Accounting Method, which could cope with larger numbers of concurrent users and offered better performance.

By the mid-1970s, the environment of sharing that had sprung up around Unix resembled the open-source movement so prevalent today. Users far and wide were enthusiastically enhancing the system, and many of their improvements were being fed back to Bell Labs for incorporation in future releases. But as Unix became more popular, AT&T's lawyers began looking harder at what various licensees were doing with their systems.

One person who caught their eye was John Lions, a computer scientist then teaching at the University of New South Wales, in Australia. In 1977, he published what was probably the most famous computing book of the time, *A Commentary on the Unix Operating System*, which contained an annotated listing of the central source code for Unix.

Unix's licensing conditions allowed for the exchange of source code, and initially, Lions's book was sold to licensees. But by 1979, AT&T's lawyers had clamped down on the book's distribution and use in academic classes. The anti-authoritarian Unix community reacted as you might expect, and samizdat copies of the book spread like wildfire. Many of us have nearly unreadable nth-generation photocopies of the original book.

End runs around AT&T's lawyers indeed became the norm—even at Bell Labs. For example, between the release of the sixth edition of Unix in 1975 and the seventh edition in 1979, Thompson collected dozens of important bug fixes to the system, coming both from within and outside of Bell Labs. He wanted these to filter out to the existing Unix user base, but the company's lawyers felt that this would constitute a form of support and balked at their release. Nevertheless, those bug fixes soon became widely distributed through unofficial channels. For instance, Lou Katz, the founding president of Usenix, received a phone call one day telling him that if he went down to a certain spot on Mountain Avenue (where Bell Labs was located) at 2 p.m., he would find something of interest. Sure enough, Katz found a magnetic tape with the bug fixes, which were rapidly in the hands of countless users.

By the end of the 1970s, Unix, which had started a decade earlier as a reaction against the loss of a comfortable programming environment, was growing like a weed throughout academia and the IT industry. Unix would flower in the early 1980s before reaching the height of its popularity in the early 1990s.

For many reasons, Unix has since given way to other commercial and noncommercial systems. But its legacy, that of an elegant, well-designed, comfortable environment for software development, lives on. In recognition of their accomplishment, Thompson and Ritchie were given the Japan Prize earlier this year, adding to a collection of honors that includes the United States' National Medal of Technology and Innovation and the Association of Computing Machinery's Turing Award. Many other, often very personal, tributes to Ritchie and his enormous influence on computing were widely shared after his death this past October.

Unix is indeed one of the most influential operating systems ever invented. Its direct descendants now number in the hundreds. On one side of the family tree are various versions of Unix proper, which began to be commercialized in the 1980s after the Bell System monopoly was broken up, freeing AT&T from the stipulations of the 1956 consent decree. On the other side are various Unix-like operating systems derived from the version of Unix developed at the University of California, Berkeley, including the one Apple uses today on its computers, OS X. I say "Unix-like" because the developers of the Berkeley Software Distribution (BSD) Unix on which these systems were based worked hard to remove all the original AT&T code so that their software and its descendants would be freely distributable.

The effectiveness of those efforts were, however, called into question when the AT&T subsidiary Unix System Laboratories filed suit against Berkeley Software Design and the Regents of the University of California in 1992 over intellectual property rights to this software. The university in turn filed a counterclaim against AT&T for breaches to the license it provided AT&T for the use of code developed at Berkeley. The ensuing legal quagmire slowed the development of free Unix-like clones, including 386BSD, which was designed for the Intel 386 chip, the CPU then found in many IBM PCs.

Had this operating system been available at the time, Linus Torvalds says he probably wouldn't have created Linux, an open-source Unix-like operating system he developed from scratch for PCs in the early 1990s. Linux has carried the Unix baton forward into the 21st century, powering a wide range of digital gadgets including wireless routers, televisions, desktop PCs, and Android smartphones. It even runs some supercomputers.

Although AT&T quickly settled its legal disputes with Berkeley Software Design and the University of California, legal wrangling over intellectual property claims to various parts of Unix and Linux have continued over the years, often involving byzantine corporate relations. By 2004, no fewer than five major lawsuits had been filed. Just this past August, a software company called the TSG Group (formerly known as the SCO Group), lost a bid in court to claim ownership of Unix copyrights that Novell had acquired when it purchased the Unix System Laboratories from AT&T in 1993.

As a programmer and Unix historian, I can't help but find all this legal sparring a bit sad. From the very start, the authors and users of Unix worked as best they could to build and share, even if that meant defying authority. That outpouring of selflessness stands in sharp contrast to the greed that has driven subsequent legal battles over the ownership of Unix.

The world of computer hardware and software moves forward startlingly fast. For IT professionals, the rapid pace of change is typically a wonderful thing. But it makes us susceptible to the loss of our own history, including important lessons from the past. To address this issue in a small way, in 1995 I started a mailing list of old-time Unix aficionados. That effort morphed into the Unix Heritage Society. Our goal is not only to save the history of Unix but also to collect and curate these old systems and, where possible, bring them back to life. With help from many talented members of this society, I was able to restore much of the old Unix software to working order, including Ritchie's first C compiler from 1972 and the first Unix system to be written in C, dating from 1973.

One holy grail that eluded us for a long time was the first edition of Unix in any form, electronic or otherwise. Then, in 2006, Al Kossow from the Computer History Museum, in Mountain View, Calif., unearthed a printed study of Unix dated 1972, which not only covered the internal workings of Unix but also included a complete assembly listing of the kernel, the main component of this operating system. This was an amazing find—like discovering an old Ford Model T collecting dust in a corner of a barn. But we didn't just want to admire the chrome work from afar. We wanted to see the thing run again.

In 2008, Tim Newsham, an independent programmer in Hawaii, and I assembled a team of like-minded Unix enthusiasts and set out to bring this ancient system back from the dead. The work was technically arduous and often frustrating, but in the end, we had a copy of the first edition of Unix running on an emulated PDP-11/20. We sent out messages announcing our success to all those we thought would be interested. Thompson, always succinct, simply replied, "Amazing." Indeed, his brainchild was amazing, and I've been happy to do what I can to make it, and the story behind it, better known.

About the Author

Warren Toomey teaches at Bond University, in Australia. He was bitten by the Unix bug in 1982 while still in high school, when he spent two weeks at the University of Wollongong learning about computers. There he encountered "an amazing system called Unix." His later discovery of software for simulating the computers that ran the earliest versions of Unix got him collecting vintage copies of this operating system.