THE RISE AND FALL OF UNIX

WHY THE BEST TECHNOLOGY SOMETIMES LOSES

Theo Pavlidis (t DOT pavlidis AT ieee DOT org)
Dept. of Computer Science
Stony Brook University

Notes for a guest lecture in a course on the History of Computing (CSE301). (Given twice, on Friday, October 13, 2004 and on Wednesday, March 9, 2005).

This version was posted on March 17, 2005.

Background

Unix was developed at Bell Labs during 1969-1971, it appeared at some Universities a few years later, and then spread like wildfire. People found Unix and the interactive shell that came with it far superior and easier to use than the existing systems, such as IBM's OS360 with its cumbersome Job Control Language (JCL). Unix provided the first widely used facility for text processing and formatting. By 1980 it was the dominant system at Universities, as well as in most technological companies. However 20 years later Unix can be found only in niche applications such as web servers and high-end graphics workstations while Microsoft Windows was the most popular operating system. What had happened?

People who have used both systems found Unix was much superior to Microsoft's systems. How did an inferior technology displace a superior one?

One short answer is that Linux (the Unix work-alike for PCs) came too late although there is more to the story.

The 1980's saw the introduction and spread of the PC, a relatively cheap hardware platform. There were a lot of things that PCs could not do, but they did enough to satisfy many users and they were cheap enough to bring computers into a mass market. By the 1990's PCs were used by millions of people for text processing, e-mail, games, and keeping track of their finances. Engineers, graphics developers, and other users of intensive computing still relied on workstations running Unix (most notably from Sun Microsystems) but the public at large could not afford those machines and had to do with the Microsoft products in spite of all their deficiencies. A parallel in the car market would be to have only two models. A minimal car selling for, say, $5,000 (think of the Yugo) and a luxury car selling for, say, $50,000 (think of a Mercedes). Most people would buy the minimal car, especially if they were to use it only for short trips to the shopping mall and if there was no used car market.

Once the large market for PCs was established, it attracted hardware and software developers so that the PC platform was becoming increasingly powerful with an increasing array of software applications available and prices dropping even more. When Microsoft introduced a decent graphical user interface with Windows 95 the price of a high end PC was about the quarter of a comparable Unix workstation and PCs started gaining additional market share. /http://www.computerhope.com/ includes a user survey that shows 91% of them using some version of Microsoft Windows, 7% using Linux, and 2% using Apple Macs.

Linux, Unix for PCs, was introduced by its creator (Linus Torvalds) only in 1991 and it was not offered commercially until 1994. By that time Microsoft's dominance was overwhelming. Why Unix for PCs was not developed 10 years earlier, before Microsoft's takeover of the market?

My own opinion is that the answer lies with the AT&T corporate culture. (**Warning:** what I have written so far I consider it to be more or less objective history. The rest is **my own personal interpretation of the course of history plus personal reminiscences** from the time of my association with Bell Labs. See the section 'About the author' at the end of this page.)

The Answer

Bell Labs was part of AT&T and had a long tradition of allowing freedom to its staff to pursue new avenues of research. This freedom allowed a bright young researcher, **Ken Thompson**, to develop a new operating system. Bell Labs folklore has it that Thompson was originally discouraged from the project; "why do we need one more operating system?" When several years later reporters asked Sam Morgan, the director of the center where Thompson was working, what was his own contribution to Unix, Morgan's answer was: "I did not kill the project." The free wheeling intellectual atmosphere of Bell Labs let other members to drop what ever else they were doing and join in the Unix development. **Dennis Ritchie** joined Thompson from the beginning and he is justly considered a co-creator of Unix. Other early contributors were Doug McIlroy and Brian Kernighan. The system went right away into use amongst the staff and kept attracting more and more contributors and more and more users in a snowball effect.

While all these exciting things were happening in Bell Labs research the corporate leadership of AT&T was slow to understand their significance. AT&T was a utility monopoly and its executives were used in a far more deliberate style of operations. Utilities are capital intensive and even if they are not legal monopolies they have little to worry about competition because of the high threshold that new companies face to enter the market. In addition AT&T kept its research branch quite separate from its development and engineering branches so technology transfer was a problem. (In contrast IBM had the policy of transferring temporarily engineers from development organizations to its research lab in order to pick up new technologies.)

A harbinger of things to come was the development of Unix for VAXes; the new 32 bit machines introduced around 1975. (The original Unix was running on 16 bit machines.) While AT&T did start a project to do the porting, they did not devote sufficient resources to the project and Unix for 32 bits was developed by Bill Joy, a graduate student at Berkeley. This came known as Berkeley Unix and an indication of AT&T's poor business strategy is that they ended up having to pay license fees to the University of California for having to use Berkeley Unix.

While AT&T created a large group to support and further develop Unix, it was part of an Engineering organization with Dilbert like management. (This is not a coincidence; Scott Adams, the creator of Dilbert, used to work for a company that was part of the AT&T system.) As a result of mismanagement, these organizations were apt to make only trivial additions to the products that came out of research ("add bells and whistles") rather than attempt real enhancements.

What was missing was an organization where people capable of high quality technical work would be properly managed (free of micro management) and would build on the results of research. A typical example, based on my direct personal experience, is the development of a text formatter that does page layout. The original Unix text formatters (troff and nroff) did text alignment only within a line because the technology of the 16 bit machines did not support the large amount of memory needed for vertical page layout. When 32 bit machines arrived researchers were reluctant (and rightly so) to embark on a project that offered relatively few new challenges. (An accessory program for helping with vertical layout was eventually written in research but, to the best of my knowledge, was never released.) Modifying the Unix text formatters to do page layout (vertical space and placement of pictures and other inserts) was the perfect project for young skilled software developers that were willing to work under the proper direction. Unfortunately AT&T did not have the framework for

supporting such efforts. On the other hand Microsoft did come with a product that does page layout although it is far more cumbersome than the Unix programs. An interesting posting on the Unix text formatter and the problems with AT&T management can be found in Mel Melchner's letter.

Another example is the development of a compiler for C++. C++ (originally called C with classes) was developed by **Bjarne Stroustrup** in the early 1980s at Bell Labs. In the early releases the program code was passed through a program that converted it into C code that was then processed by the C compiler. AT&T started a C++ compiler project, but again Dilbertian management techniques interfered, and the first C++ compiler was produced outside AT&T.

A third example of the negative effect of the AT&T corporate culture is the story of the Blit. Blit was a bitmap terminal that supported windows and interactive graphics that was developed by **Bart Locanthi** (hardware) and **Rob Pike** (software) and made internally available around 1982. The bitmap/windows technology had already been developed at Xerox PARC but it was running in stand-alone machines that were quite expensive. In contrast, Blit was meant to be inexpensive (comparable to the price of a PC) and connected to a server. While the latter feature precluded home use, it was quite appropriate for Universities and companies. Unfortunately, the AT&T management did not like the fact that the Blit used a Motorola microprocessor and insisted on redesigning it using an AT&T microprocessor. The project was given to a Dilbertian organization with the result that the introduction of the product (renamed by then) was more than two years late and priced at more than twice the original estimate. Two years might be a short time for a utility company but they are an eternity for the rapidly changing computer field.

Thus it is not surprising that no one in AT&T thought of porting Unix to the PC platform or developing the tools (especially graphic interface) that would allow naïve users to use a machine running Unix. However others did so, including Microsoft's Xenix. I cannot answer the question why Microsoft did not promote Xenix. There were several other individual efforts that did not find wide acceptance. Linux, which is a Unix work-alike system written by Torvalds did not appear until 1991 by which time the Microsoft products were well established.

Aren't the Unix people partly to blame? Not really

Is it fair to lay all the blame on AT&T? Some people will argue that there was a "Unix culture" that took pride on elegance and on building a system that professionals would love but also ignored the needs of unsophisticated users. When the psychologist Donald Norman wrote a critique of the Unix interface in 1981 [1] I remember the reaction of some Unix people. It was something like "people who have trouble with typing commands should not be using a computer." The trouble with that assessment is that there are **many more people in the world who can click a mouse button but have trouble with typing commands than people who feel at ease with typing commands**. I am giving here a very abbreviated account of the story. There has been an endless stream of debates following the Norman article, just type "Donald Norman Unix" on Google and see what comes up. A short article on the history of the "more" command is particularly worth reading in the context of a course on the history of computers.

I should also add that not everybody liked Unix and a lot has been written on the topic. A *Unix-Haters Handbook* with foreword by Norman was published in 1994. There is an online review of "Unix Haters" that will also lead you to other writings. My own view of the book is that it went overboard on several issues and in some cases it was dead wrong. I should also add that I do not agree with everything that Norman wrote, but that is not the point.

The point is that a user interface that may be great for professionals may be awful for the public at large. Creating such an interface requires different skills than creating an operating system and this is where the blame falls again on AT&T for not making an effort in that direction. Of course in those days

the idea that people would use a computer without doing any computing appeared to be strange. Ken Olsen, the founder and CEO of DEC (the No.2 computer company in the world in the 1980's, but gone since 1998) had said in 1977: "There is no reason for any individual to have a computer in his home." There is an online discussion of the Olsen quote.

Note that there was **no technical reason** why Unix could not have come up with a good graphical user interface system (rather than the horrid X Window System) or with good software developments tools. What happened is due primarily to the social and corporate factors that have been detailed above. And it is a pity because the fundamentals of Unix were quite strong.

What Microsoft did right

Bill Gates may be the Henry Ford of the computer industry. Like Ford, Gates streamlined production to make a product affordable for the masses and also tried to address the interests of the masses. And while we are on the car subject, keep in mind that cars did not become popular until features such as the electric starter and the automatic transmission became available. (Can you think of the computer counterparts of such features?) Also the importance of major corporate support for a product cannot be over-emphasized.

I want to point out two specific reasons (out of many) for Microsoft's success.

One is the organizational position of the *Program Manager* that occupies the space between the business and marketing people on one hand and the engineers and developers on the other. The Program Manager is supposed to have enough technical savvy to deal with developers and enough business savvy to make sure the developers are building something useful and to help misguided execs "see the light". If developers are left alone they are too likely to trade off usability and usefulness for things like "architectural purity", which might be intellectually pleasing but may mean nothing in the
marketplace. This point is often lost amongst the Unix/Linux people. Architectural purity and elegant code may be fine for hobbyists but not for purveyors of commodities.

Another important contribution of Microsoft is the development of tools that facilitate programming for its platforms, thus making it much easier for developers to write programs for Microsoft Windows than other systems. See an example comparing a solution using Unix tools (AWK) to one using Visual C++.

Also an illustration of the power of Visual C++ for writing computational applications.

Postscript

The inability of a corporation to take advantage of the results of its research labs has deprived the public of a superior computing environment and at the same time cost dearly to the corporation. In a five-year period AT&T lost about two billion dollars in their computer business. (To better understand that huge number, express it as the product of 20,000 times 100,000. The latter is close to the loaded annual salary of an engineer, so think of AT&T of having 20,000 engineers working for a year without any saleable output.) In 1996 AT&T spun-off Lucent and Bell Labs was part of the new company. Today both AT&T and Lucent are in trouble and Bell Labs is a shadow of its former self. The freewheeling creative atmosphere is long gone (and so are most of the people who contributed to Unix).

It was recently (early 2005) announced that AT&T is being bought by one of the former "Baby Bells." Nobody would have dreamed that in 1984 when the AT&T break up occurred. AT&T was hoping to make a lot of money by entering the computer (as opposed the telecommunication) business, but,

instead, they manage to pile up huge losses.

Unfortunately, the AT&T/Bell Labs story is not unique. Many of today's computer innovations were developed at the Xerox Palo Alto Research Center (PARC). Xerox also failed to take advantage of them and while Xerox seems to be in better shape than AT&T or Lucent, PARC exists only in name.

Is this is a coincidence? I do not think so. The free environment needed to foster innovative research can be supported only by a large corporation. Inevitably, such a corporation will have many layers of management and the top level executives are likely to be quite remote from new technologies as well as bound by a culture reflecting the original company business. You cannot teach an old dog new tricks!

Credits and Blame

I am grateful to Brian Kernighan for comments on earlier drafts of this document and to Sean Draine of Microsoft for providing me with some insight on the internal workings of the company and, in particular, the role of the program manager. I take fully the blame for anything objectionable, incorrect, or controversial in the document.

Printed References

[1] Norman, D. A. "The trouble with UNIX: The user interface is horrid", *Datamation*, 27, No. 12, 139-150.

About the author

I started using Unix around 1975 while I was on the faculty of Princeton University. In 1980 I joined the Computer Science Research Center of Bell Labs, the organization where Unix was born. I stayed there as a full time member of the technical staff until 1986 (when I joined the faculty of Stony Brook University) and as a consultant during 1986-1990. I continued as a Unix user until the late 1990's (including writing a book on the X Window System published in 1999) when I reluctantly moved to Microsoft Windows. While at Bell Labs, I worked mainly on optical character recognition, but also on graphic tools for Unix. I wrote an interactive drawing editor for the Blit (ped) and also wrote a troff preprocessor for ped files so that one could include illustrations in documents and made modification to the troff postprocessors to improve the quality of their output. These tools were part of the 8th and 9th editions of research Unix.

theopavlidis.com **Site Map**