9

Cheap time-sharing was the medium the hacker culture grew in, and for most of its lifespan the ARPAnet was primarily a network of DEC machines. The most important of these was the PDP-10, first released in 1967. The 10 remained hackerdom's favorite machine for almost fifteen years; TOPS-10 (DEC's operating system for the machine) and MACRO-10 (its assembler) are still remembered with nostalgic fondness in a great deal of slang and folklore.

MIT, though it used the same PDP-10s as everyone else, took a slightly different path; it rejected DEC's software for the PDP-10 entirely and built its own operating system, the fabled ITS.

ITS stood for "Incompatible Time-sharing System" which gives one a pretty good fix on the MIT hackers' attitude. They wanted it *their* way. Fortunately for all, MIT's people had the intelligence to match their arrogance. ITS, quirky and eccentric and occasionally buggy though it always was, hosted a brilliant series of technical innovations and still arguably holds the record as the single time-sharing system in longest continuous use.

ITS itself was written in assembler, but many ITS projects were written in the AI language LISP. LISP was far more powerful and flexible than any other language of its day; in fact, it is still a better design than most languages of *today*, 25 years later. LISP freed ITS's hackers to think in unusual and creative ways. It was a major factor in their successes, and remains one of hackerdom's favorite languages.

Excerpts from Raymond, Eric S. (1999). *The Cathedral and the Bazaar*. Cambridge, MA. O'Reilly.

10

Another important node of the culture was XEROX PARC, the famed Palo Alto Research Center. For more than a decade, from the early 1970s into the mid-1980s, PARC yielded an astonishing volume of groundbreaking hardware and software innovations. The modern mice, windows, and icons style of software interface was invented there. So were the laser printer and the local-area network; and PARC's series of D machines anticipated the powerful personal computers of the 1980s by a decade. Sadly, these prophets were without honor in their own company; so much so that it became a standard joke to describe PARC as a place characterized by developing brilliant ideas for everyone else. Their influence on hackerdom was pervasive.

The ARPAnet and the PDP-10 cultures grew in strength and variety throughout the 1970s. The facilities for electronic mailing lists that had been used to foster cooperation among continent-wide special-interest groups were increasingly also used for more social and recreational purposes. DARPA deliberately turned a blind eye to all the technically 'unauthorized' activity; it understood that the extra overhead was a small price to pay for attracting an entire generation of bright young people into the computing field.

11

Far from the bright lights of the ARPAnet, off in the wilds of New Jersey, something else had been going on since 1969 that would eventually overshadow the PDP-10 tradition. The year of ARPAnet's birth was also the year that a Bell Labs hacker named Ken Thompson invented Unix.

Thompson had been involved with the development work on a time-sharing OS called Multics, which shared common ancestry with ITS. Multics was a test-bed for some important ideas about how the complexity of an operating system could be hidden inside it, invisible to the user, and even to most programmers. The idea was to make using Multics from the outside (and programming for it!) much simpler, so that more real work could get done.

Bell Labs pulled out of the project when Multics displayed signs of bloating into an unusable white elephant (the system was later marketed commercially by Honeywell but never became a success). Ken Thompson missed the Multics environment, and began to play at implementing a mixture of its ideas and some of his own on a scavenged DEC PDP-7.

Another hacker named Dennis Ritchie invented a new language called C for use under Thompson's embryonic Unix. Like Unix, C was designed to be pleasant, unconstraining, and flexible. Interest in these tools spread at Bell Labs, and they got a boost in 1971 when Thompson and Ritchie won a bid to produce what we'd now call an office automation system for internal use there. But Thompson & Ritchie had their eye on a bigger prize.

Excerpts from Raymond, Eric S. (1999). *The Cathedral and the Bazaar*. Cambridge, MA. O'Reilly.

12

Traditionally, operating systems had been written in tight assembler to extract the absolute highest efficiency possible out of their host machines. Thompson and Ritchie were among the first to realize that hardware and compiler technology had become good enough that an entire operating system could be written in C, and by 1978 the whole environment had been successfully ported to several machines of different types.

This had never been done before, and the implications were enormous. If Unix could present the same face, the same capabilities, on machines of many different types, it could serve as a common software environment for all of them. No longer would users have to pay for complete new designs of software every time a machine went obsolete. Hackers could carry around software toolkits between different machines, rather than having to re-invent the equivalents of fire and the wheel every time.

Besides portability, Unix and C had some other important strengths. Both were constructed from a "Keep It Simple, Stupid" philosophy. A programmer could easily hold the entire logical structure of C in his head (unlike most other languages before or since) rather than needing to refer constantly to manuals; and Unix was structured as a flexible toolkit of simple programs designed to combine with each other in useful ways.

The combination proved to be adaptable to a very wide range of computing tasks, including many completely unanticipated by the designers. It spread very rapidly within AT&T, in spite of the lack of any formal support program for it. By 1980 it had spread to a large number of university and research computing sites, and thousands of hackers considered it home.