Interview with Doug McIlroy

RIK FARROW

Doug McIlroy, now an Adjunct Professor at Dartmouth, headed a small computing research department at Bell Labs. The department hired great people to do their thing, combining theory and practice. Seven of them, including Doug, have been elected to the National Academy of Engineering. UNIX was born on his watch.
doug@cs.dartmouth.edu

Rik Farrow is the editor of *;login:*.
rik@usenix.org

Over the years, I've had occasion to exchange email with Doug McIlroy. I always found Doug friendly and have long wanted to interview him.

When I finally got around to asking him, Doug anticipated that I would be interested in the role he played during the early years of UNIX and pointed me to a document he wrote in the late '80s about the research versions of UNIX [1]. The first 15 pages cover a lot of the early history of UNIX, from 1969 onward, and I really wish I had had this document when I was first researching UNIX in 1982. Doug answers a lot of questions I had then, as well as solving some mysteries that I've managed to hold on to.

The full title of this work mentions "Annotated Excerpts," and most of this document is just that: sections of early UNIX manuals. When I first encountered the UNIX manuals, reading them all was actually quite possible: there were just two volumes, perhaps a stack of paper about three inches tall (excluding the binders they were in). By the late '80s, I recall that Sun Microsystems would ship two crates of documentation about SunOS: one box full of paper and the second full of binders, perhaps 20 in all. Things have only gotten more complex since then.

But early UNIX had both a simplicity and an elegance to it that persists even to this day in the command line tools. And that's where Doug played some of his biggest roles.

Rik: I read the Research UNIX Reader, and wondered if the v8 and v9 refer to commercial versions of UNIX called System III and V? I am familiar with v6 and v7 UNIX, with most people having heard of Lions' Commentary [2], which was based on v6. And v7 became the basis for BSD UNIX.

Doug: The research and commercial systems evolved separately after v7, although not without some cross-fertilization. One more research version, v10, was documented before attention turned to Plan 9 [3]. It is a shame that only some of the good ideas of Plan 9 were adopted by the UNIX community at large. Networking would be far more transparent had Plan 9's inherently distributable architecture caught on.

Rik: You mention that you were a manager, but you were also responsible for writing some code. While most of what you wrote I don't recognize, such as your compiler-compiler (TMG), other tools would likely be familiar to command line users and script writers today, like `echo`, `tr`, and `spell`.

One thing I noticed about early UNIX tools were the short names. I used to tell people, in a joking manner, that the reason for the short names was that using Teletypes [4] for command input encouraged brevity. Even the clock daemon's name was shortened (from the prefix *chron-*) to *cron*. But I am guessing there are other reasons for short names.

Doug: Typing long names is slow on any keyboard, whether teletype or smartphone. I know of no other reason for short names. Whatever regret Ken has for quirky contractions like *creat* and *cron* is fully compensated by *grep*, a euphonious coinage so useful that it made its way into the OED as both noun and verb.

I can't help noting that `vi` commands are even shorter, and are invisible to boot—too cryptic for the taste of most of us in the UNIX room, who never strayed from `ed` until `sam` came along.

Rik: You also wrote, in the Reader, that the first shell, written by Ken Thompson and Dennis Ritchie, was very simple as it had only eight kilobytes of RAM to run in. That sounds very tough, but that limitation also seems almost unbelievable. I had more usable memory in the computer I built from a kit in 1979!

Doug: The sheer fun and productivity of UNIX inveigled lucky people to switch whatever programming they could from the megabyte address-spaces available in Bell Labs computer centers to the mere 8K on PDP-11 UNIX, and forced everyone to distill projects to their essentials [5]. Remember, though, that the 8K was backed by 16K of highly useful operating system—concentrated fare that was a far cry from today's diluted offerings. What fraction of Linux's more than 450 system calls do most users know about, much less use?

Also, 8K was much bigger back then. I just rewrote `echo`. By the time it was linked in Cygwin, its 25 machine instructions had exploded into an 8K (stripped) object file. In early UNIX, it might have been a few hundred bytes.

What bigger programs could fit in 8K bytes? The assembler, for one. Also the `roff` text-formatter—an application used by secretaries as well as researchers. And B, the ace up Ken's sleeve. This word-oriented forerunner of C produced threaded code that could run with software paging, which in particular allowed B to recompile itself.

As an aside, I remember the great sense of roominess that the 2KB memory of MIT's Whirlwind II inspired after experience with the 24-word data memory of an IBM CPC.

Rik: You also had a large role in the design of pipes, a method for joining commands, so the output of one command becomes the input to the next command. Where did the idea of the pipe come from? And wasn't the original notation different from the symbol we use today?

Doug: Pipes came out of an interest in coroutines, which had fascinated me ever since Bob McClure introduced me to Melvin Conway's concept [6]. Coroutine connections look very much like I/O. This led me to write (in a 1964 Multics document) about screwing processes together like garden hose. Joe Ossanna intended to enable reconfigurable interprocess connections in Multics, but with Bell Labs' withdrawal from Multics, I believe that did not come into use.

From time to time I toyed with (unsatisfactory) syntaxes for connecting processes in a Multics-like shell; and I repeatedly suggested that UNIX should support direct interprocess I/O. Eventually, I came up with a concrete proposal for a system call with the catchy name "pipe," and a shell syntax (exemplified by command>command>file) to exploit it. This time Ken responded, "I'll do it!"

Ken did it all in one night: creating the system call, teaching the shell to use it, and fixing programs that previously handled only named files to also deal with standard input and standard output. Pipes were an instant success. Subsequently, Ken polished the implementation by introducing the distinctive pipe symbol, "|", and revising details of the system call.

Pipes hit a design sweet spot. The world is generally unaware today (as we were then) of an earlier and more ambitious mechanism for process-to-process I/O. "Communication files" in the Dartmouth time-sharing system allowed processes to handle the entire open-file interface. They were used to implement a few multiuser services. But communication files were too arcane to make their way into programmers' mental toolkits and were never used to enable UNIX-like pipelines. In interprocess I/O, UNIX simplicity again upstaged elaborate capability.

Confession: besides fussing around for years before finding a very simple answer, I totally failed to perceive the fact that connecting processes via pipes is logically more powerful than via stored serial files. You can replace an intermediate file with a pipe, but not always vice versa. An interactive session, such as `dc|speak` (a talking desk calculator), won't work if it has to treasure up all the output of `dc` before running speak. Bob Morris pointed this out on the very day pipes first worked. Had Ken and I been conscious of it, UNIX might have gotten some pipelike facility—perhaps not so simple—much earlier.

Resources

[1] D. M. McIlroy, "A Research UNIX Reader": https://archive.org/details/a_research_unix_reader.

[2] J. Lions, *A Commentary on the Sixth Edition UNIX Operating System,* 1977 (out of print): http://www.lemis.com/grog/Documentation/Lions/book.pdf.

[3] Rob Pike, Dave Presotto, Sean Dorward, Bob Flandrena, Ken Thompson, Howard Trickey, and Phil Winterbottom, "Plan 9 from Bell Labs," *Computing Systems,* vol. 8, no. 3, Summer 1995: https://www.usenix.org/legacy/publications/compsystems/1995/sum_pike.pdf.

[4] ASR 33 Teletype Information: http://www.pdp8.net/asr33/asr33.shtml.

[5] Gerard Holzmann, "Code Inflation," *IEEE Software,* March/April 2015: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7057573.

[6] Melvin E. Conway, "Design of a Separable Transition-Diagram Compiler," *Communications of the ACM,* vol. 6, no. 7, July 1963, pp. 396–408.