

Unix at 50: How the OS that powered smartphones started from failure



Enlarge / Ken Thompson (sitting) and Dennis Ritchie (standing) in front of a PDP-11. Ritchie annotated this press image for Bell Labs as “an amusing photo,” and he joked that he had much “more luxuriant and darker hair” at the time of the photo than when it appeared in magazines like the March 1999 Scientific American (which, unfortunately, incorrectly swapped IDs for the two).

reader comments

127 with 88 posters participating

Share this story

Maybe its pervasiveness has long obscured its origins. But Unix, the operating system that in one derivative or another powers nearly all smartphones sold worldwide, was born 50 years ago from the failure of an ambitious project that involved titans like Bell Labs, GE, and MIT. Largely the brainchild of a few programmers at Bell Labs, the unlikely story of Unix begins with a meeting on the top floor of an otherwise unremarkable annex at the sprawling Bell Labs complex in Murray Hill, New Jersey.

It was a bright, cold Monday, the last day of March 1969, and the computer sciences department was hosting distinguished guests: Bill Baker, a Bell Labs vice president, and Ed David, the director of research. Baker was about to pull the plug on Multics (a condensed form of MULTiplexed Information and Computing Service), a software project that the computer sciences department had been working on for four years. Multics was two years overdue, way over budget, and functional only in the loosest possible understanding of the term.

Trying to put the best spin possible on what was clearly an abject failure, Baker gave a speech in which he claimed that Bell Labs had accomplished everything it was trying to accomplish in Multics and that they no longer needed to work on the project. As Berk Tague, a staffer present at the meeting, later told Princeton University, "Like Vietnam, he declared victory and got out of Multics."

Within the department, this announcement was hardly unexpected. The programmers were acutely aware of the various issues with both the scope of the project and the computer they had been asked to build it for.

Still, it was something to work on, and as long as Bell Labs was working on Multics, they would also have a \$7 million mainframe computer to play around with in their spare time.

Dennis Ritchie, one of the programmers working on Multics, later said they all felt some stake in the success of the project, even though they knew the odds of that success were exceedingly remote.

Cancellation of Multics meant the end of the only project that the programmers in the Computer science department had to work on—and it also meant the loss of the only computer in the Computer science department. After the GE 645 mainframe was taken apart and hauled off, the computer science department's resources were reduced to little more than office supplies and a few terminals.

As Ken Thompson, another programmer working on the project, wryly observed for the Unix Oral History project, “Our personal way of life was going to go much more spartan.”

Luckily for computer enthusiasts, constraint can at times lead to immense creativity. And so the most influential operating system ever written was not funded by venture capitalists, and the people who wrote it didn't become billionaires because of it. Unix came about because Bell Labs hired smart people and gave them the freedom to amuse themselves, trusting that their projects would be useful more often than not. Before Unix, researchers at Bell Labs had already invented the transistor and the laser, as well as any number of innovations in computer graphics, speech synthesis, and speech recognition.

Make way for Multics

Multics had started off hopefully enough, although even at first glance its goals were a bit vaguely stated and somewhat extravagant.

A collaboration involving GE, MIT, and Bell Labs, Multics was promoted as a project that would turn computing power into something as easy to access as electricity or phone service. Bell Labs researchers would have a jack in their office that

would connect their terminal to the Multics mainframe, and they would be able to access—in real time—the mainframe's entire resources. They would also be able to store files on the mainframe and retrieve them at will.

If all this sounds incredibly trivial, it's evidence of how important these features rapidly became—even for simple computing tasks. But when Multics was first conceived in the early '60s, file storage was a novelty, and “time sharing” (or the ability for multiple users to share access to a single computer's resources) had only been done experimentally, not in a production environment with a high number of users.

Computers in the early 1960s ran programs one at a time, one after the other. A researcher at Bell Labs would write a program, convert it into whatever form of input the computer accepted (punch cards, paper tape, or magnetic media for really fancy machines), and drop it off at the computer center. A computer operator would queue up the program, run it, and then deliver the printed results and the original program to the researcher.

If there was a mistake in the code, the hassle of printing out punch cards, taking them down to the computer center, and then waiting around for results was rewarded with a printout saying something like “SYNTAX ERROR.” Perhaps you might also get a line reference or some other possibly helpful information.

As programs became more complicated, this method of debugging code became even more frustrating than it already was. But no company or university, even Bell Labs, was in a position to buy a mainframe for each individual researcher—in 1965, the GE 645 that Bell Labs used to develop Multics cost almost as much as a Boeing 737.

Thus, there was widespread interest in time sharing, which allowed multiple researchers to run programs on the mainframe at the same time, getting results immediately on their remote

terminals. With time sharing, the programs weren't printed off on punch cards, they were written and stored on the mainframe. In theory, researchers could write, edit, and run their programs on the fly and without leaving their offices. Multics was conceived with that goal in mind. It kicked off in 1964 and had an initial delivery deadline of 1967.

MIT, where a primitive time-sharing system called CTSS had already been developed and was in use, would provide the specs, GE would provide the hardware, and GE and Bell Labs would split the programming tasks.

<https://arstechnica.com/?p=1489117>