Linux

Vergazov Artem, 11.18.21, 7:50PM

In this essay, I am going to write about the rise of Linux as a freely distributed system, try to analyze why this rise was possible, and discuss the pros and cons of it for the programming community.

First, a few words about Linux. Linux is a kernel for open-source Unix-like operating systems (OS). Contrary to the popular belief, Linux is not the whole OS, the term “kernel” means it is just the core of the OS, the part that handles interactions between hardware and software components of the computer. It is usually a part of what’s called “a GNU/Linux distribution” a combination of Linux as the core and the GNU software. Linux has become the largest open-source software project in the world. Professional and hobbyist programmers from around the world contribute to the Linux kernel, adding features, finding and fixing bugs and security flaws, and providing new ideas — all while sharing their contributions back to the community. The source code may be used, modified and distributed commercially or non-commercially by anyone under the terms of its respective licenses, such as the GNU General Public License (GPL). Now, to understand how Linux became so popular, let us get back to the origins of Linux and GNU.

The roots of Linux go back to the creation of Unix at AT&T’s Bell Labs in 1969 by one of Bell Labs researchers Ken Thompson. Unix became very popular among universities because at the time the majority of the university computer science departments ran PDP-11 machines and Unix was designed for PDP-11, too (not initially though, at the start they did it for PDP-7 but made an upgrade to PDP-11 after the first success of Unix). Also, Unix was shipped with the complete source code which gave people the opportunity to experiment with it freely. So, Unix quickly filled the void of scientific computing machines. With the growth of variety of computers, it became hard to have to rewrite Assembler code for each type of machine and Thompson came up the high-level language B that was supposed to take care of all the machine-dependent instructions. Due to B’s lack of structures and other weaknesses, this attempt was not successful but B gave birth to the C language, a successor to B, designed by Dennis Ritchie. This language remains actual even now. Thompson, Ritchie and company continued the development of Unix until it became so popular that even other companies started trying to develop their own version of Unix. In 1984, AT&T was broken up by the US government and shortly thereafter released its first commercial Unix product, System III.

So, for decades Unix was the standard operating system for commercial computing, but there was a catch. It was owned by AT&T, and it only ran on high-end equipment. Programmers wanted something they could tinker with on their personal computers.

In 1984, Richard Stallman started working on GNU, a Unix-clone that stands for "GNU's not Unix." By 1991, Stallman and company had successfully rewritten most of Unix, but they were missing one crucial component: the kernel, which is the fundamental core of an operating system---the part that talks to the hardware and translates the basic input from your keyboard, mouse, and touchscreen into something the software can understand. So, a Finnish computer science student named Linus Torvalds announced a new project. Torvalds decided to create a kernel. Linux rapidly grew in size and evolved into a full, production UNIX clone, as virtual memory, a more sophisticated file system, and many other features were added. Although it originally ran only on the 386 (and even had embedded 386 assembly code in the middle of C procedures), it was quickly ported to other platforms and now runs on a wide variety of machines, just as UNIX does.

And, just as in the GNU’s philosophy, Torvalds kept it open-source and free distributed. Everyone could play around with the code, set up their own distributive that would correspond to their needs and share it with the community. And, what is even more important, the programmers could suggest their improvements to the code, their bugfixes and other helpful stuff because of the Linux’s open-sourceness. The whole world could support the development thus making Linux true “for programmers by programmers” software. That is why popularity of Linux has risen so rapidly.

In my opinion, this an excellent example of how large software should be development – the ideas always have to be shared with the community. The programmers should never hide the code behind the license, instead they should share it freely and let the community help with the development because the community is full of people willing and able to help. The results will be much better than if only one team was doing the development. And Linux is the example of that.

One last thing to mention is that during the early days of Linux existence, a lucky coincidence took place that may be one of the reasons of Linux popularity boost. In 1992, Berkeley, running out of funding, decided to terminate BSD (Berkeley Software Distribution) development with one final release, 4.4BSD (which later formed the basis of FreeBSD). Since this version contained essentially no AT&T code, Berkeley issued the software under an open source license (not GPL) that let everybody do whatever they wanted except one thing — sue the University of California. The AT&T immediately sued the University of Carolina for copyright and trademark infringement. This case kept FreeBSD off the market long enough for Linux to get well established. Had the lawsuit not happened, starting around 1993 there would have been competition between two free, open source UNIX systems and the question whether Linux would win this battle stays open.