The Linux Operating System

Rowin Andruscavage

May 5, 1998

ENGRG 298 Spring 1998

Contents

Introduction

The recent anti-trust action taken against computer industry giant Microsoft brings to light the question of "when does an operating system become a public resource, like the electrical or telephone systems?" What many people are unaware of is the existence of an operating system that is already a public resource, and the remarkable story behind its development that defies conventional wisdom.

1 Background

In the early years of computing technology, AT&T's Bell Laboratories codeveloped the first versions of the Unix[1] operating system with the University of California at Berkeley. When the system became functional, Bell Laboratories copyrighted the software, not allowing anyone access to the source code, and began licensing only binary machine level distributions of the core Unix components for use in industry. This move aggravated the academic community, which was interested in further developing the platform. The inner workings of their computer science experiments were always obscured by the nebulous "black box" of the Unix operating system, which handled (and sometimes mishandled) every interaction with their programs. Without access to the source code which AT&T guarded, researchers were often unable to trace the cause of problems with their work.

The academic community soon realized that they would have to reinvent the wheel in order to have access to the inner workings of their computers. To this end, a few societies were formed in the mid-80's to completely redevelop the Unix system from scratch and thus free themselves from their dependency on the closed AT&T licensing. These organizations, such as the Free Software Foundation based in Cambridge and the FreeBSD project in Berkeley, managed to produce clones of the Unix operating system and all of its utilities based on open standards such as POSIX.[2]This time, however, they took special care to define legal safeguards that prevented private corporations from placing copyrights and patents on their software. These kinds of licenses, called public licenses or "copylefts," aimed to keep their work and all derivative work from falling under the exclusive control of companies that might inhibit further development.

Meanwhile, a Finnish graduate student named Linus Torvalds facing the same licensing problems through his research also began developing an operating system. He started by extending upon a decrepit operating system meant for instructional demonstration called MINIX. He placed the source code for the core component of his operating system, or kernel, on the internet for public scrutiny, and eventually

[1] One of the few names that is not an acronym, but actually a play on words of Multics (Multiplexed Information and Computing Service). Multics was one of the first timesharing operating systems, apparently conceived of in terms of telephone network multiplexes. UNIX was initially a single-user system, hence the name. Ironically, it has kept the name even after it had developed timesharing.

[2] (Portable Operating System Interface for Computer Environment Spec 1170)

a band of students, professors, and freelance programmers joined him in tinkering with and developing the kernel. They named it Linux and decided to license it under the General Public License[3] to protect its continued growth. The kernel was coupled with the work from the Free Software Foundation in order to form a complete operating system. The first widely-distributed version was released in 1991, and it has developed rapidly since then. The installed Linux system base is beginning to rival that of Apple's Macintosh, but unlike Apple's, is growing.

2 The Heroic Act of Invention

Much of the success of the Linux operating system is attributed to its creator Linus Torvalds. He is a programmer like everyone else on the development team, but is generally regarded as the leader by other developers. Torvalds provides a strong central authority to guide Linux development. He establishes the direction, pace, and organization of the worldwide development effort.

Guidance

Although many developers are free to work on whatever aspect of kernel programming they are interested in, most look to Torvalds for defining the priorities. This was more important during the earlier stages of the kernel development, when key decisions needed to be made and big changes meant rewriting a lot of previously working code. If Torvalds backed a movement towards a completely different but more powerful file system, for example, more people would be willing to rewrite their individual contributions to support the new feature. They would not feel so compelled to revise their work after a suggestion from a peer. Torvalds fills this role because he has built up a strong trust from other developers. However, he states that any one of a number of people in the development team also have sufficient trust to take over as leader of the project if necessary.[4] Linux should be able to

[3]The General Public License is not exactly public domain, but is more like a copyright against further copyrights and patents. The source code to software covered by this license must be made available by anyone who copies or modifies it for further distribution.

[4]Torvalds had the following to say on this issue :

> The day people think linux would be better served by somebody else (FSF being the natural alternative), I'll "abdicate". I don't think that it's something people have to worry about right now - I don't see it happening in the near future. I enjoy doing linux, even though it does mean some work, and I haven't gotten any complaints (some almost timid reminders about a patch I have forgotten or ignored,

outlast the lifetime of any one person. In the meantime, Torvalds provides the single authority figure that unifies and focuses the direction of the development effort. His presence not only prevents Linux from fragmenting into several Linux derivatives, like FreeBSD[5] has, but also allows it to dexterously change momentum to adapt to new technological trends.

Another important function that Torvalds performs is the pacing of the development cycle. In order to be successful as a product, most programs have to go through alternating periods of development and debugging. During the development phase, new features are added, which may conflict and cause problems with existing code. At appropriate times during the cycle, Torvalds announces a "code freeze," during which all of the developers stop trying to add new features and focus on fixing all of the conflicts between parts of their code. Once they are happy with the way everything is working, they will officially release a "stable" version of Linux intended for general use. The pace at which this process occurs is somewhat critical. Although many Linux users don't mind the risks involved with using the unstable development versions, people who need reliable computer systems will install the most sophisticated stable operating system they can find at the time. Since Linux has a faster development cycle than other operating systems, the current stable version of Linux is more likely to be the most advanced operating system to choose at the time. In the 7 years since Linux development has started, Torvalds has called for code freezes four times. The kernel usually stabilizes within a matter of months.

The Linux Development Model

Torvalds's most interesting role by far is the innovative way in which he organizes the development team. His organizational style is best summarized by Eric Raymond in his paper "The Cathedral and the Bazaar," which compares the Linux development model with conventional software engineering. The philosophy goes

> but nothing negative so far).
>
> Don't take the above to mean that I'll stop the day somebody complains: I'm thick-skinned (Lasu, who is reading this over my shoulder commented that "thick-HEADED is closer to the truth") enough to take some abuse. If I weren't, I'd have stopped developing linux the day ast ridiculed me on c.o.minix. What I mean is just that while linux has been my baby so far, I don't want to stand in the way if people want to make something better of it.

(Taken from Linus's reply to someone worried about the future of Linux)

[5] FreeBSD is actually one of a series of loose derivatives of 386BSD, along with NetBSD and OpenBSD.

"release early and often, delegate everything you can, be open to the point of promiscuity." Torvalds maximizes the amount of human work spent on Linux by closely involving users and developers with the work. New versions of the development Linux kernels are released at least once a week, which gives people a good feeling for the rapid progress. This helps make anyone who contributes to the effort feel intimately involved, especially when they see a fix they submit become part of the kernel the very next day. Torvalds is as much of a personality as he is a programmer, using his charisma to woo people into Linux development. He allows almost anyone to contribute code, since he knows that they will then have an emotional bond to Linux development and put even more time into it. Torvalds takes very little credit for the relatively small amount of code that he contributes to Linux, but actively dishes out credit to other developers in order to tap more of their volunteer effort. He is definitely not the sole inventor of Linux, and most of the copyrights of the source code are actually owned by other people.

Torvalds's personality reflects his interest in streamlining Linux development. He describes himself as lazy, which is the reason he delegates most of the work to other people. This delegation is key to the success of a large development effort, however. Most software development teams are kept small, like the "Cathedral" in Eric Raymond's paper. This is because the amount of interaction necessary between developers to work on part of a program increases exponentially as the number of developers increases. It seems like a miracle that something as complex as an operating system would still work after being braided together in "Bazaar" fashion by literally hundreds of programmers. This is made feasible since most of the developer interaction occurs with Torvalds and not necessarily with every other developer. Torvalds knows that he could get more work done by recruiting a dozen people for a couple of projects rather than working on them himself. Additionally, there is strong proof that a team of moderately talented programmers can produce higher-quality code, whereas even an expert programmer could get deluded by some intricacy of the software design.

His organizational style is reminiscent of emerging trends in management, such as increasing the personal involvement between employees and the company. This is almost a necessity with free software development, since most Linux developers

do not receive any salary for their work. Linux development also has a resemblance to the concurrent engineering process that competitive corporations are adapting. Traditionally, products would iterate through the distinct phases of conception, design, development, production, marketing, and customer feedback during the typical product cycle. Many corporations that produce evolutionary products, such as automobiles, now use concurrent engineering to speed up product design. Concurrent engineering involves increasing feedback between each of the design phases by having them all collaborate fully on their project simultaneously rather than in turn. Corporations use this technique to reduce costs, but Linux has used it to speed development. Since the computer science industry is still a relatively young field of engineering, it can still be slow to adapt to established strategies that don't seem to make sense in their field. However, this provides a good example of how new techniques in organization can be thought of in terms of old technologies.

The advantages of these principles behind Linux development become clear in comparison with FreeBSD, another freely-redistributable Unix clone.[6] FreeBSD is in some ways technologically superior to Linux, and also has a more relaxed license. However, FreeBSD lacks a central authority figure, in favor of a more traditional team of programmers in charge of a pool of developers. The BSD license allows companies to modify FreeBSD source code and copyright the new code as their own. This may cause the source code to become closed and proprietary, which is exactly what the General Public License protects against. However, this license policy may have hurt FreeBSD development, as many developers hold the principles behind the General Public License with an almost religious fervor.

[6] Torvalds himself would beg to differ :

> "Other than the fact Linux has a cool name, could someone explain why I should use Linux over BSD?"
>
> > "No. That's it. The cool name, that is. We worked very hard on creating a name that would appeal to the majority of people, and it certainly paid off: thousands of people are using linux just to be able > to say "OS/2? Hah. I've got Linux. What a cool name". 386BSD made the mistake of putting a lot of numbers and weird abbreviations into the name, and is scaring away a lot of people just because it sounds too technical."
>
> (Linus Torvalds' follow-up to a question about Linux)

3 The Social Process of Invention

Linux has a social structure just as peculiar as its development structure. Linux differs from commercial operating systems in essentially a similar fashion that public radio differs from commercial broadcast radio. Public radio rejects corporate endorsements in order to offer an objective and unbiased program to its audience. Linux is a public operating system in many ways, in that it looks to its user base for support in lieu of corporate contributions. Although this preserves its focus on quality and technology, the Linux system has somewhat alienated the corporate sector.

Since the development of Linux is so open, the users and the engineers of Linux have tended to be the same people. The free communication made possible by the internet allows users to interact with the development team fast and efficiently. This allows Linux to adapt almost instantaneously to users' needs. The past provides countless examples of Linux kernels being fixed within hours of someone on the internet finding a problem. The Linux and FreeBSD teams often compete to see who can fix a newly-discovered problem in their respective kernels first. Both had incorporated a patch to fix a bug caused by the Intel Pentium processor within four hours of its discovery, before Intel even acknowledged that the bug existed. In contrast, almost all of commercial operating systems did not respond to the problem for several weeks afterwards. The free interaction among the users and the developers of Linux helps it progress at a rapid pace.

Commercial Dissatisfaction

Linux is still very much in the hobbyist stage of social acceptance. The computer industry has been somewhat slow to acknowledge Linux as a competitive computer platform, despite its wide use. Companies are reluctant to create Linux versions of their programs for the same reason no one wanted to invest in radio broadcasting in the early 1920's : they couldn't see how to make money from it. A few commercial corporations have embraced Linux by selling computer servers, operating system packages, and user support focused on it. However, the majority of information technology companies choose not to use Linux or make Linux versions of their programs available, citing such reasons as a lack of demand, no central marketing

drive, and no official product support. These conceptions of Linux mostly arise from miscommunication to or misunderstanding by corporate management, who unfortunately have the power to make technical decisions for companies.

One of the problems with the Linux distribution system is that it is virtually impossible to keep accurate records of how many people are using it. Many individuals register with various "Linux counters" on the internet, which puts the user base at about 6 million. However, many institutions do not openly admit to using Linux. This usually comes about because a corporation's management decides to pay for and install a commercial computer server at the suggestion of a marketing salesperson disguised as an information technology consultant. The regular computer staff at the company then secretly replaces the commercial operating system with Linux, since it tends to work better and is thus easier to maintain. This happens at surprisingly many companies, even large ones like Boeing. The generally accepted number of Linux computers in the world is 10 million, partially based on computer programs that poll the internet. Due to the wide uncertainty, computer companies tend to neglect the the demand for Linux software and hardware support.

Even though Linux manages to grow by virtue and by word-of-mouth without relying on expensive marketing campaigns, many commercial companies deny the viability of a product without a brand name behind it. This is not so much an issue in Europe and Japan, where merchandise is rated more on quality and not on advertisement, but is central to the American concept of the market economy and mass distribution systems. Only a handful of U.S. enterprises have started to market and sell Linux software packages – complete automated systems that are ready to perform diverse tasks – but names like Caldera, Red Hat, and SUSE possibly have not rung as big as a single unified corporation selling Linux would. Many endeavors are turned away by the mention of "free software," thinking that money cannot be made. However, the word "free" refers to freedom of the source code and not to price. The free software community recently began referring to their wares as "open source software" in an effort to encourage more companies to develop under this model.

Ironically, the lack of a centralized marketing department is one of the Linux development model's greatest advantages. Linux is a technological system at heart,

and benefits from not having close ties with a marketing group to put impositions on or try to control the kernel development. When Torvalds graduated and took a job in the U.S., he purposefully avoided working with a company that had a strong commercial interest in Linux. Doing so might have allowed the company, such as Red Hat, to influence kernel development to favor their particular Linux package. Keeping separate ties with commercial marketing systems has allowed Linux development to remain entrenched in technological and not economic issues.

The last reservation that commercial industry has against adapting to Linux is the lack of a centralized support structure. Companies want an organization to go to for technical support or to take responsibility for a problem. Linux mostly has relied on the USENET internet newsgroups for questions and answers since its inception. Most companies don't realize the effectiveness of these newsgroups for providing detailed troubleshooting instructions. People can post questions about problems they are having with Linux, and usually receive about half a dozen replies from others who have successfully dealt with those problems within 24 hours of posting. The newsgroups are also frequented by the developers of the Linux kernel themselves, which means that real problems with the kernel will usually go on to get fixed. All of these people put an unaccountable amount of work into providing these solutions on essentially a volunteer basis. Companies don't expect such high reliability and responsibility from volunteers, but that is how things have been in practice. Most corporations, on the other hand, have comparatively poorer quality and slower technical support in practice. Also, many companies specifically disclaim any responsibility to fix their products. Linux will be at a disadvantage as long as people fail to realize these benefits over commercial products.

These limitations are beginning to be lifted as Linux is mentioned in the mainstream media more and more often. Since Linux doesn't have any significant advertising campaigns, it must rely on public press to provide exposure. Acknowledgements such as *InfoWorld's* "1997 Best Technical Support" award or coverage by National Public Radio put Linux in a positive light as a viable alternative operating system. The mass media is a double-edged, sword, though. For every positive article, there are equally as many articles from magazines serving special commercial interests, such as MSNBC and ZDNet. Even though these attempt to downplay

Linux's advantages, they nevertheless help spread the word about its very existence.

4 Hughes's Network Model of the System

The evolution of Linux can be said to fit into the phases of system development outlined by Thomas Hughes in *Networks of Power*. Because Linux is still young and maturing, its ultimate path through the phases has yet to be determined. However, Hughes's model may give us a few indications as to where it might go in the years ahead.

Invention and Definition of a System

The Linux Operating System is something of a misnomer. Linux-based computers actually run the GNU[7] Operating System, the fruit of the Free Software Foundation's effort at rewriting Unix from scratch. Linux consists of merely the kernel, which is a relatively small but fundamental part of a working computer. The kernel acts as the nervous system of the computer, directly controlling hardware such as keyboards, disk drives, and even other computers on its network. It also controls how a computer divides its time and resources between programs that are running. Programs that want to interact with the hardware, such as by writing a file to the hard disk, must do so through the kernel. In Linux's case, the kernel polices the programs to prevent them from performing illegal operations that may jeopardize the system or violate the operations of other programs. Linux's technical success is gauged by the speed and effectiveness with which it performs this task.

Linux is a system because it provides a standard interface to the hardware of a computer. Since new standards are extremely difficult to develop and adapt, the interface was modelled after an established existing system, Unix. Unix was quite expensive at the time, and only ran on special workstation hardware, not on personal computers. Therefore, there was quite a niche for Linux, dubbed as "Unix for PC's."

Linux is an open system, progressing very much at the whim of its users/developers. When Torvalds put his kernel on the internet, he was astonished by the amount of attention it received by other developers. He had never planned on getting such a

[7] GNU's Not Unix, an example of a "recursive" acronym appreciated by computer programmers

large response, or on the kernel growing into a fully-fledged operating system in so short a time. He had just been trying to run basic GNU programs on his home PC. The users were instrumental in defining the technology as an alternative operating system.

Technology Transfer

The development of the Linux kernel probably could never have progressed far without the widespread availability of internet access. The majority of kernel developers are all "well-connected" to the internet, frequenting discussion groups and keeping in touch through email. Many still have not met each other in person to this day. However, the internet has not only allowed people to transfer technology to and from remote places in the world, such as Finland, but also to collaborate on it.

For example, an elevator company in Japan was interested in using Linux to make an intelligent embedded controller. Unfortunately, the way in which Linux shares processor time between running programs may cause the program controlling the elevator to be stopped for several milliseconds to give time to another program. This would be unacceptable, so the company improved the realtime support of the kernel and contributed their work back to the development effort. Technology transfer evidently moves both ways in this case.

System Growth

As the system came close to stabilizing for the first time, people other than hackers and college students began to use Linux. It built a reputation as the operating system with the fastest networking, finding use as web servers and network filesystems. The more it grew, though, the more apparent its reverse salients became.

A lot of kernel development goes into writing device drivers for specific or new hardware. Since Linux isn't a mainstream platform, companies don't find it economical to produce drivers for a market that they would make relatively little money on. This hasn't been a big problem in the past, since drivers written by the kernel development team often work better than the company-provided driver for the mainstream operating system. However, the hardware companies are sometimes reluctant to release the detailed specifications of their products necessary for creating

drivers. This presents a problem, since the drivers may need to be painstakingly reverse-engineered, or cannot be made at all. As a result, Linux has at times been unable to work on certain hardware, which limits the number of computers that Linux can successfully work on. In response to this critical problem, the Linux user community has been forming advocacy groups to try to enlist the cooperation of the companies, either through shows of support for Linux or boycotts of the company. As a result, many companies have become more open with providing hardware specifications.

Humorously, some advocates have been too vocal or downright unpolite, which has necessitated a new trend of "advocacy control" within the Linux user community. Overzealous advocates have caused the downfall of at least one other platform, the Amiga, and Linux users are afraid the same type of thing may happen to them. On the whole, the Linux user community has proven adept at gracefully redefining the direction of technologies without insulting the companies involved.

Linux has recently been adapting a new role as a desktop operating system instead of solely a server operating system. A significant critical problem slowing its spread is the lack of an easy-to-use graphical user interface. Unix is notoriously unintuitive in its use of many obscure text commands, and Linux is no exception. Linux began to become more accessible by incorporating the XFree86 graphical display system from a non-profit company in Texas. XFree86 Inc. had been working on extending the GUI developed by Xerox and MIT onto personal computers. Even though Linux then had a graphical display, it still didn't have an interface that was simple to use. In response, the mostly-German K Desktop Environment project was started in 1997 to provide a powerful and attractive desktop that did not require its users to have special experience with Unix. Unfortunately, the project is not covered under the General Public License and might run the risk of becoming proprietary in the future. To that effect, free software developers (or rather open source developers, as they are now called), learning from experience, have already begun GNOME[8] to proactively avoid that situation. The user interface problem will likely be solved by both of these projects in the near future.

Currently the most critical problem has been to gain more support from the

[8]GNU Network Object Model Environment

commercial software industry. Although the kernel and the operating system have been developed completely based on the principles of free software, the programs that run on it do not necessarily need to be non-commercial as well. Migration is a difficult task, however, especially when the system people are migrating from has significant momentum in another direction. A handful of programs called emulators have been developed to help people make the transition. These allow Linux users to run some programs intended for Macintosh, DOS, or Windows computers, and they occasionally work better than they did under their native platform. Although originally just a temporary solution to this critical problem, emulation may be the preferred method of programming in the future. The Java programming language is essentially a universal language for a completely emulated platform. If it becomes successful, Linux will likely follow in its success due to its powerful emulation capabilities.

Momentum

Even if Linux does not become a successful competitor in the personal computing market, it will still have its niche in mission-critical and specialized computer systems. The availability of its complete source code is a rare feature not shared by any other operating system of its sophistication. This allows researchers and engineers to know everything about how and why the operating system works the way it does. In space shuttle experiments or even just elevators, the engineer needs to be able to predict how and why a computer can fail, which is obscured by not having the source code. Linux's source code availability also allows engineers to build one-of-a-kind computer systems with a modified version of the kernel. These systems include supercomputers made from clusters of Linux PC's tuned for their task of number crunching, such as the Beowulf cluster at NASA and the cluster made by Digital Domain for rendering scenes of ships sinking in *Titanic*.

However, most of Linux's momentum comes by virtue of its reliability, performance, and economics. When Linux is installed on a computer, it usually stays there, assuming the user has the experience and patience to deal with Unix. Information technologists can relate many stories of companies that running Linux servers that decide to try using Windows NT. Most convert back to Linux within a

week due to reliability and performance issues. Linux also excels in economical factors, since for little or no money, most distributions can provide services equivalent to at least $5000 of commercial software. It would be difficult at best to undermine this type of competitive advantage.

Furthermore, since the source code to is freely available, the operating system has been ported to work on many more architectures. Initially intended for Intel PC's, Linux now works on every major hardware platform, from Macintosh's PowerPC and Motorola 68K platforms to Digital's Alphas to Sun's SPARCstation. Developers are also making versions of Linux for more esoteric hardware, such as the PalmPilot handheld computer. Torvalds has joked that the ultimate goal of the Linux project was "complete world domination." Although the propagation of kernels may seem to point towards this goal, the variety assures that the project is not bound to any one particular architecture that may become obsolete or die out in the near future. For this reason, the momentum behind Linux will likely continue a fair distance into the future.

5 Conclusion

The success of the Linux operating system has proven a principle that Richard Stallman, the founder of the Free Software Foundation, has been arguing for decades. Stallman believes that software does not constitute artifacts, since it costs virtually nothing to reproduce and distribute. Rather, software is a form of intellectual property that should be shared with the rest of the world and not shielded in proprietary secrecy. He has led a long crusade to encourage companies to always publish the source code to their programs in order to make them easier to develop and maintain. Corporations have always argued back that they would never be able to produce quality programs of any complexity if thousands of people worldwide were meddling with them. The Linux phenomenon has more or less proved Stallman right.

Netscape Software has decided to attempt to apply the Linux development model to their Mozilla web browser product, and has released its source code. This will be the first test to see whether Stallman's idealistic concept is commercially viable. If they succeed, they may revolutionize the software industry. If they fail, the open source development model may be shunned by commercial software companies for

decades to come.

References

[1] Charles, Dan, "OS by Committee", *All Things Considered*, National Public Radio, April 8 1998; http://www.npr.org/ramfiles/980408.atc.14.ram

[2] Dieter, George E., *Engineering Design*; McGraw-Hill Inc., 1991

[3] Foster, Ed, "1997 Product of the Year : Best Technical Support Award" *Infoworld*, 1998; http://www.infoworld.com/

[4] FreeBSD Inc.; http://www.freebsd.org/

[5] Hughes, Thomas P., *Networks of Power*, The Johns Hopkins University Press, 1983

[6] Kirch, John, "Microsoft Windows NT Server 4.0 versus UNIX", May 3, 1998; http://www.kirch.net/unix-nt.html

[7] Markon, Sandor; Sasaki, Kenji; "Linux for Embedded Systems", *Linux Journal*, **41**:66

[8] Raymond, Eric S., "The Cathedral and the Bazaar", 1998; http://sagan.earthspace.net/~esr/writings/cathedral-bazaar/

[9] Ritchie, Dennis M., "The Evolution of the Unix Time-sharing System", *Bell System Technical Journal* **63**:8, Oct 1984

[10] "The GNU General Public License", Free Software Foundation, Inc., Cambridge, MA; 1991; http://sunsite.unc.edu/mdw/LDP/gs/app-gpl/node1.html

[11] USENET comp.os.linux.development.system

[12] Vizard, Michael; "Linux Torvalds Talks Economics and Operating Systems"; Infoworld April 9, 1998; http://www.infoworld.com/cgi-bin/displayStory.pl?/interviews/980409torvalds.htm

[13] Welsh, Matt, "Linux Installation and Getting Started", Specialized Systems Consultants, Inc., Seattle, WA; 1996; http://sunsite.unc.edu/mdw/LDP/gs/gs.html