The Linux Ecosystem: an Actor-Network Analysis of Free and Open Source Software

A Research Paper submitted to the Department of Engineering and Society

Presented to the Faculty of the School of Engineering and Applied Science

University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree

Bachelor of Science, School of Engineering

Talis Basham

Spring 2020

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Advisor

Kathryn A. Neeley, Associate Professor of STS, Department of Engineering and Society

Introduction

Free and Open Source Software, hereafter referred to as open source software is software that is distributed under a license that permits it to be freely redistributed. In some schemes known as copyleft it requires the modified version to be redistributed under the original terms. Open source software is in use in almost all sectors of technology. It is easy to take for granted the availability of high quality software ecosystems consisting of programs and support. Although there have been major open source projects from corporate creators in recent years, many of these projects were started and initially developed by individuals.

The Linux kernel is one such high profile case. What once was the hobby project of a Finnish graduate student has grown into a formidable product with market shaping implications developed by thousands of contributors. Linux can be found in a wide variety of computers from the pervasive Android cell-phones to the world's top 500 most powerful publicly recognized supercomputers. It is apparent that inspecting the process behind this type of explosive growth is key to understanding the factors that contribute to the success of an open source project.

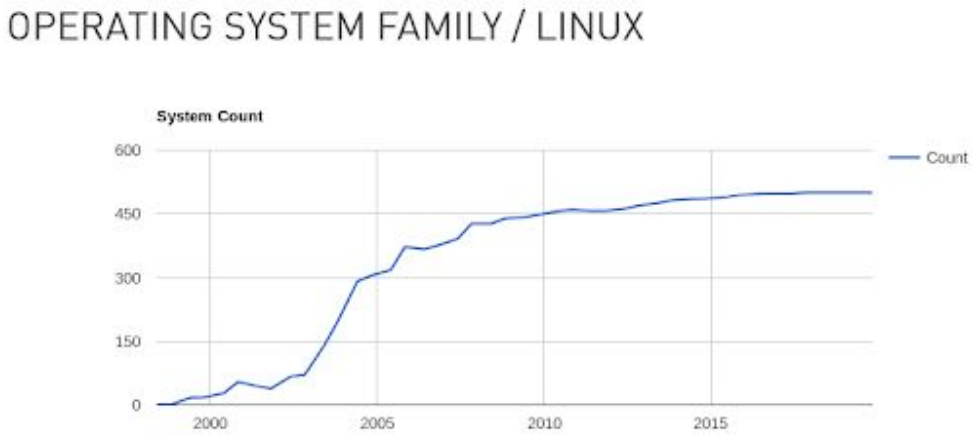

Figure 1: Linux share in Top 500 Supercomputers (Top500 2020)

Although the history of Linux's development is well documented, it is not clear what contributed to its success. It's development is popularly associated with the heroic effort of lone hackers working passionately on a feature they desire, but the romantic sentiment is contrasted by the reality of the modern development community in which core contributors are primarily paid to develop for Linux at their jobs. In this paper, I argue that Linux's growth was the result of Torvald's effective mobilization of the preexistent free software community and the aggressively competitive edge afforded by freely licensing. Towards this aim, I connect the concept of a software ecosystem to actor-network theory (ANT) explore the history of Linux's development through the lens of ANT.

A Brief History of The Linux Community

It is important to make clear the scope of the Linux project. Linux is an operating system kernel: the piece of software that interacts directly with and abstracts away the hardware. The programs that a user interacts with, including interface, are known as the userland.

The history of Linux's development is well documented because it is an open source project and all contributions are public. Linus and other hobbyist developers drove the initial development of Linux. However, by the late 90s the majority of patches have come from developers working at large information technology companies which use Linux. This widespread adoption took advantage of highly competitive, free, licensing and the familiarity of university computer science graduates during the explosive growth of the internet with the POSIX paradigm.

Linus Torvalds is the original creator of Linux, and he retains final say in conflicts arising in development. He is responsible for determining what additions are accepted into the official

version of Linux. This follows a common leadership model in the open source software world: the Benevolent Dictator for Life. The BDFL model stands to be contrasted against a more designed by committee method other projects take (Raymond 2000).

Important early support came from the existing free software movement and the GNU project. GNU's goal is to provide an open source operating system, consisting of an entire UNIX like set of tools and programs. However, the GNU project's kernel was not ready for use at the time of Linux's development. The tools that the GNU project had already written well complemented the Linux kernel and were readily adopted by it. Linus Torvalds mentioned porting the GNU C Compiler in his post on Usenet first announcing Linux (Torvalds 1991).

As the Linux kernel grew in size and usage, bigger companies began to take interest and contribute to the project. For example, Intel and IBM provided support for using their CPUs. Many companies using Linux in their products choose to submit their modifications to Linux for inclusion in the official version because it is easier to maintain them if they are accepted.

Many groups provide their own distributions of Linux bundled with additional software. Some companies provide commercial support for their distribution. Red Hat is one such company that provides professional support for Linux installations. Red Hat employs developers to write not only patches for the Linux Kernel, but also develops user space programs such as SystemD and Pulseaudio. Conversely, there are distributions maintained entirely by volunteers such as Debian.

In the present day upwards of 80% of changes to the Linux Kernel are written by people paid by their employer to provide. However, Linux has a reputation for being a community driven effort, and is often presented as a success of non-commercial development. Linus

Torvalds's (1991) original vision was a clone of the operating system MINIX he created as a hobby project to familiarize himself with the i386 architecture. His interest in Linux grew as the problem morphed into that of creating a usable and efficient operating system kernel. Torvalds increased his dedication to work on Linux as it gained increasing adoption .

The developers coalesced around the Linux kernel with the intention of making it more usable for their purposes. I identify this push towards the bettering of Linux for industry use as the problematization central to the actor network which I will explore in my final thesis. Contributions most immediately benefited the contributor, but many of them were widely useful and improved the quality of Linux for other purposes as well. Contributor's interest in improving the common ground drove the formation of connections between the aforementioned actors.

The early Linux community created the first Linux distributions which provided simple setup and management tools for Linux systems. These attracted new users by lowering the barrier to entry and greatly benefited the growth of the Linux community. One such early distribution, RedHat, was purchased by a software distributor.

In the nineties Linux experienced significant improvements to stability for information technology industry use. It was freed from its single platform constraints in 1995 when it was ported to DEC Alpha and SPARC, two non-Intel processor architectures common in Unix based workstations and servers of the era. This was followed by other changes which had the result of making Linux more desirable to information technology applications. In 1998 this desirability was confirmed by IBM's commitment to provide significant funding to Linux development. Gaetano termed this arrangement of open source contribution by processor manufacturers "hardware-software complementarity" (Gaetano).

Application of Actor-Network Theory to Software Ecosystems

Software ecosystem is a nebulous term that refers to the resources available for utilizing a particular piece of software (Jansen). The term appears primarily in business and systems engineering contexts. The activity and size of an ecosystem are implicitly linked with the health of the primary software. For an operating system, the user space programs comprise the majority of the software ecosystem.

Actor network theory provides a useful model for understanding the two way influences between society and technology modelled upon the formation and changes of networks around a central problem (Stanforth 1994). ANT lends itself well towards analysis of a complex and growing ecosystem of heterogeneous factors that a large scale open source project provides. Stanforth surveys and condenses Callon's work developing ANT. I will use ANT to explore the Linux development process and explain the working

ANT provides a translational model of power in which power is held by those directing the changes in the central feature of the network (Stanforth 1994). This translation corresponds to the extension of the Linux development network and describes the the shifts in power within it. Stanforth (1994) summarizes the four central "moments of translation" from Callon's work:

- "Problematization"—the principal actors (the researchers) make themselves indispensable to the other entities (fishermen, scientific community, scallops) by defining the nature of the problem and forcing the others to accept a way forward (the research program);

•“Interessement”—the principal actors lock the others into place by interposing themselves and defining the linkages between the others (the research program becomes the recognized obligatory point of passage between the global and the local);

•“Enrollment”—the principal actors define the roles that are to be played and the way in which the others will relate to one another within these networks; and

•“Mobilization”—the principal actors borrow the force of their passive agent allies and turn themselves into their representatives or Spokespeople.

Callon (1984) makes it explicit that these occur continuously, and in no specific order. Over time new actors may join the network and old ones leave as the central goal changes based on the translation of power within the system. The dynamism modeled by this process provides an interesting model for the exploration of the growth of a heterogeneous system.

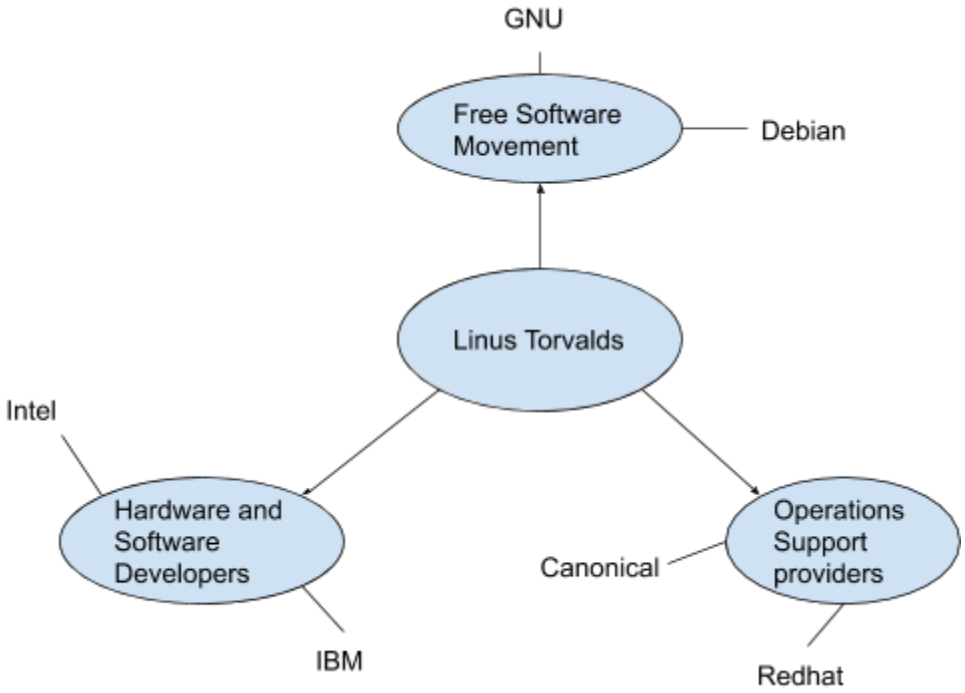

Figure 2: Major Groups of Linux Actor-Network with Examples

Raymond (2000) identifies the defining feature of open source development as the way in which useful programs are shaped to further the needs of individuals and groups who have found them useful. This accounts for the groups introduced in the previous section. Raymond's model trivially maps to the human and technological actors of an actor-network model with translational power. However, it originates from a mindset of virtuosity ethics and does not necessarily account for the other sociotechnical actors. In this paper, I explore the history of Linux with an actor-network model to investigate scaling in open source development. The individuals and groups mentioned within the brief history above provide many of the actors that I use as evidence. I also explore the influence of other sociological conditions such as the state of the market.

Translation of Power within the Linux Kernel Development Network

The actors I choose to explore are the primary groups of contributors to the kernel present in the above history. Linus Torvalds warrants exploration as an actor separate from the rest of the developers because he provides a final point of passage for development.

I identify the push towards the bettering of Linux for use as the problematization central to the actor-network. Contributions most immediately benefited the contributor, but many of them were widely useful and improved the quality of Linux for other purposes as well. This interest in improving the common ground drove the formation of connections between the aforementioned actors. I explore the growth of connections of these actors as the interessement of the network. The diverse groups are connected primarily as a hub around the source of the Linux kernel, although some carry relationships with the other actors. The communication required to develop this took place through the kernel mailing lists. I identify the enrolment of

the ANT framework with the actual development of software and activities supplemental to it. The most defining feature of open source software is that it is constantly changing. It is common for older software to be supplanted by a competing better software. In the Linux user space there have been many replacements of old software with the new. This can cause friction between actors. For example the GNU userland can be entirely replaced with other alternatives. I identify the mobilisation of the network with the act of development. The contributors to the kernel send in their relevant modifications to the kernel development community where they are reviewed through a strict process. However, some patches are developed and distributed outside this structure. The grsecurity patches provide an example of this. They provide a comprehensive set of patches intended to enhance security, which in 2017 they made available only to their paying customers.

Linus takes a very personal involvement in the kernel development, and he continues to have final authority on decisions, making him an inevitable point of passage for the system. He is known to write vitriolic emails in context of technical matters. Around the time grsecurity was being accused of violating the GPL he described their patches as “pure garbage” and lambasted them for not developing their patches within the typical system. He made similarly disparaging comments about Intel’s patches following the recent discovery of the “spectre” hardware exploit. A source of conflict is ideological: the GNU project’s free software community proscribes software that is not compatible with their license, the GPL. The GNU project’s ideology manifests in stringent requirements for recommendation of Linux distributions. They took offense to the partnership of Canonical and Amazon, and raised awareness of it. However, this dedication is commonly acknowledged to represent only a fringe part of the Linux community.

Conclusion

The Linux ecosystem's development exhibits a translation of power away from Torvalds, towards a community of individuals, and then to a community of stronger business interests. Over the decades of coverage the real power dynamics of the network grew mismatched from public perception. The GNU project initially influenced its development towards the production of the missing component of a fully free operating system that they required and provided a mature userspace. Applying ANT to Linux's development suggests that its most successful feature was the continuous momentum enrolling larger and more powerful groups of actors.

The result of this paper is an application of ANT to software ecosystems and open source projects in particular through the case study of Linux. My research demonstrates that the software ecosystems spoken of in business literature can be explored as sociotechnical systems. This model can be used to judge the development of other projects and evaluate their potential for future usefulness.

Resources

Bond, H. S. (2005). What's So Great About Nothing? The GNU General Public License and the Zero-Price-Fixing Problem. *Michigan Law Review*, 104(3), 547 - 571.

Callon, M. (1984). Some Elements of a Sociology of Translation: Domestication of the Scallops and the Fishermen of St Brieuc Bay. *Sociological Review*, 196 - 233.

Elder-Vass, D. (2015). The Moral Economy of Digital Gifts. *International Journal of Social Quality*, 5(1), 35 - 50.

Gaetano, L. (2015). A Model of Corporate Donations to Open Source Under Hardware–software Complementarity, *Industrial and Corporate Change*, 24(1), 163 - 190.

How the development process works (n.d.). Retrieved from https://www.kernel.org/doc/html/v4.15/process/2.Process.html

Jansen, S. (2014). Measuring the Health of Open Source Software Ecosystems: Beyond the Scope of Project Health. *Information and Software Technology*, 56.

Lee, G. K., & Cole, R. E. (2003). From a Firm-Based to a Community-Based Model of Knowledge

Pacey, A. (1983) Innovative dialogue. In The Culture of Technology Creation: The Case of the Linux Kernel Development. *Organization Science*, 14(6), 633 - 649.

Raymond, E. S. (2000, August 24). Homesteading on the Noosphere. Retrieved from http://www.catb.org/~esr/writings/cathedral-bazaar/homesteading/.

Torvalds, L. (1991, August 25). What would you like to see most in minix? Retrieved from comp.os.minix.

West, J., & Dedrick, J. (2001). Open Source Standardization: The Rise of Linux in the Network Era. *Knowledge, Technology & Policy*, 14(2), 88 - 112.