David Tucker

Gerald Moulds

CMPE 185

21 March 2013

Bazaar Governance

Technology will soon pervade every aspect of a person's life. This is clear from the explosion of relentless innovation surrounding the Internet in recent years. It is also quite clear that the world will likely change the way it is governed as the largest communications network in history continues to evolve. So far, the Internet has remained open and relatively unregulated, and numerous projects have flourished because of it. Many of those projects have adopted a development model that ensures the open environment that allowed them such success is preserved. These methodologies, matured by Free/Libre and Open Source Software (FLOSS) movements, could greatly benefit all involved if applied appropriately to government throughout its extended adaptation of technology. To best understand how to do this, the values and advantages of the free software movement must be acknowledged.

A Brief History of FLOSS

Considering the youth of modern computers, it comes as no surprise that the origin of FLOSS is deeply intertwined with the history of operating systems which are some of the most key pieces of software in existence. Their importance comes from their ability to facilitate the running of other software in parallel. They abstract user programs away from the hardware they run on and coordinate which processes get to use limited hardware

resources that all software must share (e.g. processor, memory, storage, etc.). It makes sense that software with such a pivotal task would be among the first written. In fact, one of the very first large software development projects to reach maturity was an operating system known as UNIX.

A team of Bell Labs researchers began the UNIX project in 1969. Initially spelled Unics (UNiplexed Information and Computing Service), the project was heavily influenced by its predecessor, Multics, which was written in assembly language. Three years later, UNIX underwent a critical change when it was abstracted from assembly language to the newly introduced C programming language. Not only did this crucial step make porting UNIX builds to other machines easier, but it showed that important and useful software could be run efficiently despite being compiled from a higher-level computer language. This innovation helped spur interest in UNIX. AT&T (Bell Labs' parent company) then licensed it to many others, and it, in turn, became very popular and prolific. As more people got their hands on its source code, expanded derivative systems began to appear. For example, UC Berkeley created the Berkeley Software Distribution (BSD) in 1977 which made significant progress in computer networking and eventually went on to influence SunOS and Mac OS X among others. Another UNIX-inspired derivative, MINIX, has been used for educational purposes since the late 1980s and has since had a large influence on most UNIX-like systems in existence today. Ultimately, though, UNIX was still a proprietary platform which meant that its development would be dictated in the best interest of AT&T.

By 1983, a man named Richard Stallman saw the need for a universally accessible operating system. There were a few alternatives to UNIX, but none that were licensed in a

way that he believed respected users' rights to the software. He began laying plans for a free operating system he would make called GNU (which recursively expands to "GNU is Not Unix") and published a manifesto of the project detailing what would later become the Free Software Foundation's (FSF) definition of free software. While he did not actually found the FSF until 1985, the GNU Project operated on the same principles that the FSF mandates free programs offer:

0. Freedom to run the program for any purpose
1. Freedom to study how the program works and change it
2. Freedom to redistribute copies
3. Freedom to improve the program and release your improvements to the public

(Stallman)

These principles prescribe a practice known as copyleft in which licensors offer their work on the terms that redistributions and derivative works be disseminated on the same terms that they were originally offered under. In other words, if Bob gives Alice a copy of his program under a copyleft license, Alice may only redistribute it (modified or not) under a compatible copyleft license. At the time that the FSF was founded, however, no such license existed in widespread use.

So, in 1989, the FSF introduced the GNU General Public License version 1 (GPL). The GNU Project releases all the software it produces under this license which has caused the GPL to become more popular as the project and the free software movement have gained momentum. The GNU GPL has since undergone two revisions. Although it has a

concise development history, the story of its influence is anything but.

By 1992, the GNU Project had almost completed its entirely free operating system; however, it was still missing one of the most crucial pieces: the kernel. This component is directly responsible for controlling machine hardware and interfacing with software. In a way, it supports the rest of the operating system. That year a Finnish college student named Linus Torvalds released a kernel he had been working on under the GNU GPL, and by the next year, many were using it to fill the final void in the GNU operating system. He had created and somewhat modeled his kernel on MINIX, the operating system he had spent time studying at the University of Helsinki. Torvalds named it Linux, and so, GNU/Linux distributions were born. Linux, though, remained the buzzword around the completion of the world's first free operating system, and so many people mistakenly call entire distributions simply Linux.

GNU/Linux systems are now deployed around the world in perhaps every industry, including supercomputing. Throughout the 1990s and early 2000s, developers created hundreds of distinct distributions geared towards all types of applications. Free software has remained an important topic in the Linux community, not only because it is released as such, but because of its great success on the laurels of the GNU Project. As more and more people started participating in GNU/Linux projects, it became more common for outsiders and newcomers to mistake free software as synonymous with gratis (free of charge) software. This brought about a minor stigma that products of the free software movement are cheap or low quality. To combat this notion, developers often use the term libre ("free" in Spanish and French) as a less ambiguous substitute, especially in software

naming. Luckily, this subtle misconception did not seem to deter many developers before being formally addressed (and somewhat rectified) by the open source movement.

Over the years, the Linux kernel continued to grow (from about 10,000 lines of code) and mature (to nearly 15 million lines) while giving a huge boost to the free software movement. Unfortunately, free/libre software had not enjoyed quite the same success among commercial institutions. This is because one of the primary avenues for creating profit from software development involves keeping the software's source code closed to the public and selling precompiled binary copies at a premium. The best business model proponents of the free software movement have discovered as a replacement suggests that commercial software companies make money by charging for services and support while offering their software gratis. A few companies found success doing just that, notably Red Hat. Red Hat Enterprise Linux (RHEL) costs anywhere from $50 to $18,000 and has become a de facto standard in many commercial environments despite its mostly free licensing. It is used particularly heavily in industrial servers very much like its source-clone CentOS which is available free of charge online. Not surprisingly, though, this model has not drawn as much interest among most software vendors.

In an attempt to make free/libre software more appealing to companies, Bruce Perens and Eric Raymond founded the Open Source Initiative (OSI) in 1998. They chose the name open source as a clearer, more business-friendly alternative to free software. What makes the open source movement unique is that it is accompanied by a development model that has proven to be rather effective. Raymond articulated this model in a well-known essay he wrote a year earlier titled *The Cathedral and the Bazaar.* The

essay likens the traditional, centralized methods to the construction of a cathedral "carefully crafted by individual wizards or small bands of mages working in splendid isolation." In contrast, he compares the more decentralized open source model to “a great babbling bazaar of differing agendas and approaches” (2). Early releases of new projects, extensive collaboration among user and developer, frequent integration of new code, different builds for general use or testing, source code modularity, and strategic oversight characterize this so-called bazaar. The bazaar-style open source movement embraces many of the same ideals as the free software movement; however, according to Richard Stallman, the open source model is not quite the same as free software. He believe the difference lies in purpose. The free software movement focuses on the philosophically ethical way to utilize software while the open source movement is more geared toward integrating the practical utility of free software into real world projects. For many, open source acts as a happy medium between truly free and proprietary development methodologies.

Today, free/libre and open source software projects are abundant and flourishing, including examples like Apache, Mozilla, LibreOffice, Eclipse, FileZilla, and Debian to name a few. The unprecedented reach of these projects is astonishing. Over 50% of all websites on the internet today use the Apache HTTP Server. It is available under the Apache Software license which was introduced in 2004 and has since become a prominent and compatible alternative to the GPL. Apache is not the first development project to create a custom free license. In 1999, the Berkeley Software Distribution’s license was broadened to the less restrictive (and very creatively titled) Modified BSD license which is often used for its extreme brevity and GPL compatibility. Another example

is the Mozilla Foundation which makes Firefox, one of the top three most used web browsers in the world, and offers it under a Modified BSD and GPL license hybrid called the Mozilla Public License (MPL). Perhaps one of the strongest free software communities, though, remains GPL-based. The Debian Project began in 1993 as one of the earliest GNU/Linux distributions. It has been dubbed “The Universal Operating System” because of its extensive ability to adapt to almost any hardware configuration available. Debian has thousands of contributors and is the origin of more than 180 derivative distributions including the massively popular Ubuntu. It is governed by the Debian Social Contract which preserves it as community-based FLOSS and encourages collaboration for common benefit. With such large and vibrant user and developer bases, it is not hard to see the advantages projects like Debian enjoy over analogous proprietary softwares.

Advantages of Peer Review

One of the most obvious benefits of more users is more source code review. Eric Raymond once said, “given enough eyeballs, all bugs are shallow” (9). He, of course, refers to the peer review process where any user (peer editor) may find and expose design or implementation flaws. Similar processes are used in literary establishments and academic journals. This has a highly beneficial effect on the security of committed code. Having a potentially infinite number of peer editors exposes the software to many people with different areas of expertise and diverse backgrounds. In many ways, users “attack” the application from all angles, innocuously or otherwise. But having more attackers invariably suggests the presence of many complementary defenders in such a community. This means not only are inevitable code vulnerabilities more likely to be found faster, but they

are also more likely to be repaired sooner. And since “attacker” and “defender” are not particularly defined roles, anyone who uses the software can potentially be either or both. Even the United States Department of Defense (DoD) has acknowledged these benefits in reliability and particularly security. A report to the DoD published in 2003 regarding a study of FLOSS recommended the adoption of libreware in many areas within the DoD’s scope. It suggested that

> The DoD should develop generic policies both to promote broader and more effective use of FOSS, and to encourage the use of commercial products that work well with FOSS... For Security, use of GPL within groups with well-defined security boundaries should be encouraged to promote faster, more locally autonomous responses to cyber threats[, and] for Research the policies should encourage appropriate use of FOSS both to share and publish basic research, and to encourage faster commercial innovation (Bollinger, 3).

At the end of the day, though, the most important point to note about the open source development model is that it focuses on the best interest of the software’s users which produces more useful and relevant features. This, in turn, adds to the perpetuity and relevance of the project and, effectually, the community as a whole. The best part is that the community is unrestricted. If Bob does not like the direction a software project is heading, he can simply split (fork) the project and create whatever he likes. If his fork turns out to offer more useful or desirable features, it can be reunited with or even replace the main branch at any time. Much like capitalistic competition, this (quite literal) pull and push of

development spurs innovation. So much so, in fact, that this kind of open development has evolved from a software-specific movement into a general purpose algorithm engineering methodology. This step may seem subtle, but it is a cornerstone of the confidence placed in online communications today.

The modern Internet consists of many interconnected computers (nodes) which rely on a set of protocols (algorithms) to communicate. In particular, there are two types of protocols that have benefitted from a generalized FLOSS-like development model. The first type, cryptographic protocols, allows users to transmit sensitive or private information securely over unreliable or insecure mediums. Mathematicians made this possible by devising methods of encrypting (cloaking) data to be sent and decrypting received data so recipients may understand it again. Anyone even nominally versed in network security or encryption knows that the strength of these algorithms comes from the open nature of their specification. The National Institute of Standards and Technology (NIST) also acknowledges this: “System security should not depend on the secrecy of the implementation or its components” (United States, 2-4). In particular, the Transport Layer Security (TLS)[1] protocol uses the Advanced Encryption Standard (AES), which has been published by NIST, and RSA, which has been publicly available since its patent (held by MIT) expired in 2000. Secure communication, however, is but an afterthought to communication itself which leads to the second type: communication protocols. These flourish in open, decentralized environments like the Internet, because of their affinity to inspire indiscriminate collaboration. For instance, the Transmission Control Protocol

[1] The more widely known Secure Sockets Layer (SSL) predates TLS.

(TCP), which assures reliable data delivery between nodes on the Internet, is used by most if not all nodes on the internet today. Since its publication in 1974, developers have tweaked and improved the protocol through three updates and many extensions to the point that, now, many of the inefficiencies and holes in the initial specification have been patched and battle tested by the public. The reason protocol engineers use similar, unrestricted models is because the bazaar approach to collaboration Raymond describes applies constant, unbiased peer review to new features and old components alike. Ironically, the infrastructure supported by these protocols has only facilitated more collaboration while becoming the most useful peer editing tool in existence. This, in turn, makes it easier to improve each other's work and perpetuate the bazaar cycle through still more peer editing and so on. In fact, the Internet has become the most powerful, useful, and revolutionary tool available today, and its widespread success stems from its FLOSS-like beginnings and the decentralized, "bazaar" environment it grew up in.

Revolutionary Innovation

The evolution of the Internet began with the creation of ARPANET around the same time that the UNIX project started in 1969. It was funded by the US DoD as a project for cutting-edge network research. Like UNIX, ARPANET inspired others and had derivative and comparative systems built after it; however, unlike UNIX, the advances made by others could more easily be shared back upstream. In particular, progress was discussed through a collaboration between the American Advanced Research Projects Agency (ARPA) and administrators of European networks, the International Network Working Group (INWG)[2].

[2]Later renamed to the International Federation of Information Processing Working Group (IFIP WG) 6.1

According to Vinton Cerf, an Internet pioneer, “the effort at developing the Internet Protocols was international from the beginning” (Hauben, 1). One member, Louis Pouzin (designer of the French CYCLADES network), and his colleague realized that centralizing the responsibility of reliability within the network is not as effective as using a decentralized model in which hosts take responsibility for the implementation of most desired solutions themselves. This displacement of duty is called end-to-end communication.

By the time the working group produced its first such solution (TCP) in 1973, ARPANET had gained new developers and expanded from four nodes to around fifty. As it expanded, it also became more diverse. Independent network researchers each pursued the most efficient protocols and practices, and many (particularly Americans and Europeans) ended up solving similar problems in different ways. For instance, around the same time ARPANET got underway, many Europeans had begun experimenting with their own virtual circuits (as opposed to packet-switching) based networks which used a model similar to the Plain Old Telephone System (POTS). They called their technology X.25, and it operated very differently from ARPA’s networks. When people develop in such isolated (localized), cathedral-like environments, the solutions that emerge can prove to be incompatible together. In the case of early-seventies internets, inharmonious link layer protocols (which are responsible for coordinating local traffic within the network) experienced this unfortunate side effect. The question of how to organize and coordinate diverse remote nodes frustrated engineers (and world leaders alike), until INWG blessed the world’s infant networks with the Internet Protocol (IP) in 1974. With IP, the European X.25 virtual-circuit and the packet-switching networks like ARPANET became cohesive.

TCP/IP together gave the Internet significant momentum in its youth because of its ability to adapt to any scheme in use by a given autonomous network. Instead of worrying about connecting different hosts residing on different networks, researchers used abstraction to hide the link layer. Analogous to what C did for UNIX, TCP/IP made joining diverse networks much simpler by connecting networks and letting each network worry about how to get messages from its gateway to the appropriate hosts within it. This shift to a more decentralized, autonomous model mimics Raymond's bazaar and, thus, met great success. This is one of the reasons why ARPANET directly evolved into the Internet while UNIX practically (but not technically) required a replacement (i.e. GNU) before having the true impact of its technology realized.

Before becoming the Internet of today, though, the DoD transferred responsibility for the administration of ARPANET, which had become fully operational, to the National Science Foundation. ARPA had accomplished its goal of pioneering a state of the art network, and others built networks like it all over the nation. The NSF joined ARPANET within its own (mostly research-based) network, NSFNET which connected many early participants, including the Department of Energy's (DOE) Energy Sciences Network (ESNet), the National Aeronautics and Space Administration's (NASA) Science Network (NSN), and the NSF's own Computer Science Network (CSNET). Integration of digital network technology spread quickly within government and research institutions. Its utility superseded the application to national defense that spawned it, and it spread to many disciplines.

Standards

Historically, any form of expansion the Internet experienced caused isolated pioneers to explore new frontiers. Before the fruits of their labor could make way for better, newer, and more advanced frontiers, some form of open decentralization and autonomy had to be introduced. It also had to be done in such a way as to preserve the coordinated efforts that facilitated the innovation. This trend suggests that people explore decentralized technologies and develop new solutions with it. When the various projects that result are joined in an open way, a new decentralized environment is created, and the process can repeat which allows for sustainability.

According to this trend, the plethora of networks that constituted NSFNET needed open unification to evolve. Like the choice faced by America's Founding Fathers, the role of the unifier had to be crafted with care for the environment to survive and thrive. Instating one with too much authority risked creating an over-centralized cathedral out of the Internet, but one with too little authority would not unify users or inspire progress. It turns out that just the right authority manifested in the Internet Engineering Task Force (IETF) which first convened in 1986. It has proven to be a perfectly adequate solution through its system of standardization and voluntary adoption. Standards establish open, cooperative solutions for problems many people have in common. When people widely adopt open standards, common ground can be easily established quickly and by anyone using them, even complete strangers. For example, consider the protocols discussed earlier, communications and cryptographic. With the standardization of those two classes alone, any user on the Internet may communicate instantly and securely with anyone in the world connected to the same network and using the same standards. The IETF publishes

standards in the form of memos called Requests For Comments (RFCs) which are never modified, only made obsolete. Adoption of IETF standards is completely optional, but undeniable benefits await those who do. This encourages all participants to want to adopt standards voluntarily which perpetuates the use of standards and creates a sustainable form of governance for Internet protocols and policies. Of course, in a decentralized model, the IETF cannot be the only authoritative organization regarding the Internet. It collaborates with other standards-producing entities to divide and conquer all the different areas where unification can encourage cohesion. In a sense, these standards essentially function as open source protocols, and they have transformed the user experience of the Internet forever.

World Wide Web

A user's interaction with the Internet today is quite media-centric. Many interactive tools exists that can do anything from creating music to administering social networks. This facet of the 'net began to show prominence in the early 1990s with the shift from logistics to content; however, the Internet required massive expansion to attain the scale, reach, power, and diversity it enjoys today. The technology that would facilitate this growth manifested in two parts. The first allowed visual representation of programs and data through a desktop environment known as a graphic user interface (GUI). GUIs had already been introduced by the time the second surfaced in 1991 at the European Organization for Nuclear Research (CERN) in Geneva, Switzerland. Its creator, Tim Berners-Lee, called it the World Wide Web (WWW). It introduced hypertext processing on three levels: the HyperText Transfer Protocol (HTTP), an accompanying server (httpd) and client

(WorldWideWeb), and the HyperText Markup Language (HTML) which allows dynamic web pages to be built and parsed with ease. Berners-Lee explains “HyperText [as] a way to link and access information of various kinds as a web of nodes in which the user can browse at will” (1). Basically, it facilitates the discovery of new or related information through the extensive use of links to all kinds of multimedia. With the completion of the GNU free operating system and the rise of Microsoft shortly thereafter, everything for the ultimate international bazaar that is the modern, commercialized Internet had fallen into place. All the collaboration and trust that characterized the birth of the Internet began infecting the world at large. Bill Gates, founder of Microsoft, assimilated the widespread ramifications of the Internet to a tidal wave, saying “The Internet is at the forefront of all this [innovation,[3]] and development on the Internet over the next several year will set the course of our industry for a long time to come” (1). Berners-Lee envisioned this as well. To further support progress toward a sustainable, decentralized, and unrestricted web culture, he founded the World Wide Web Consortium (W3C). Like the IETF, W3C aims “to promote the widest adoption of Web standards” through the use of “Recommendations that can be implemented on a Royalty-Free (RF) basis” (W3C). This helped cultivate an independent, decentralized, and bazaar mindset within Internet culture. As a result, the Web has enjoyed unbelievable growth.

Continued Growth

Since the birth of the WWW, the Internet has been expanding exponentially (see figure 1), and it is expected to continue. It has already invaded coffee shops and strip

[3] Regarding video-dominated multimedia and low-cost communication

malls, penetrated homes and businesses, and attached to pockets and desks everywhere. As this global resource grows, people will continue to integrate it in all aspects of modern life. Meanwhile, the WWW has evolved from a primitive linking mechanism into what many now know as Web 2.0.

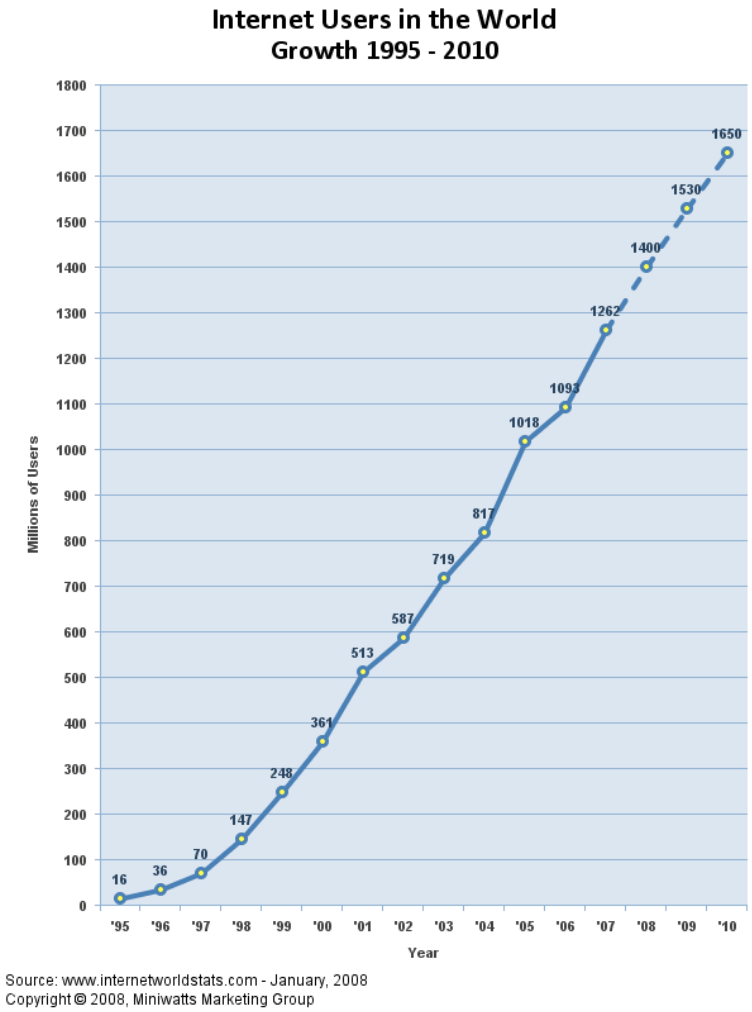

Figure 1. Internet Growth 1995-2007

Web 2.0 represents a shift in focus from basic resource sharing to quality content presentation and in web culture from building for the Internet to building for its users. This has manifested in a surge of hundreds of thousands of new and useful applications that do

everything from simulating 3-dimensional environments to storing and manipulating personal data. In fact, the surge in programs than manage and organize personal data has forged a new frontier called cloud computing. Some pioneering commercial organizations have recently begun to offer centralized ("cloud-based") data management for the higher reliability of load-balanced data centers and portability of synchronized devices. The Internet's ability to produce new frontiers faster and more abundantly than ever before is one interesting effect of the exponential growth on the Internet that make the future development it will support hard to predict and plan for. One thing that is certain, though, is the approach of technological omnipresence. The most blatant evidence of this lies in the prospect of the so-called "Internet of Things," a vision of Internet applications using sensor arrays to monitor and control everyday items remotely or automatically. The Internet of Things embodies the bazaar environment that sustains the Internet community, but also has significant potential for abuse and maladaptation due to its direct interaction with more than the cyber world. It illustrates the not-so-subtle human desire to incorporate the real world with the Web.

Into the Internet

Social networks serve as a perfect case study when considering the effects of technological integration with the real world. They are just as centralized services as other new frontiers like cloud computing, but they have immediate real world effects during the course of their regular operation in a very decentralized way. Even though social networks are so new, people have immersed themselves in them at an unprecedented rate. It started with MySpace around 2005 which allows users to create a personal profile to represent

their online presence. The profile can be customized using HTML or one of a plethora of templates and generators that have appeared in the MySpace community. Users interact by commenting on each other's profile or exchanging private messages. Photographs and videos can be embedded within a profile using media hosting sites like Photobucket and ImageShack. Indeed, MySpace has had a very Web-centric take on social networking. This endeavour actually failed to retain its dominant status, though, in favor of the now widely known Facebook which lacks much of the customizable profile features. Facebook uses a simpler, more standardized way of signifying the online presence of a user. The triumph of Facebook over MySpace can also be attributed to a difference in philosophy. According to business growth expert Adam Hartung points out that the brilliance of [founder] Mark Zuckerberg was his willingness to allow Facebook to go wherever the market wanted it" (1). Zuckerberg and others built Facebook built to be a more people-centric digital tool. Facebook overcame MySpace because users wanted a more real, people-centric experience and less technical involvement.

Five years later Facebook remains as relevant as ever with more than one billion monthly active users. It has led social media to become some of the first public tools on the Web to influence real events in real time. This became most obvious in late 2010 when Facebook, Twitter, YouTube, and other social services played a central role in shaping political debates in the Arab Spring. Conversations about revolution often preceded major events on the ground, and social media carried inspiring stories of protest across international borders" (Howard, 1). People are discovering they can use technology to interact with government. They can assemble and coordinate themselves, and report

anything that occurs to the world instantly. This kind of social organization is clearly here to stay, and it is only a matter of time before modern governments must answer the call of technology to keep up with the needs of the people and to remain relevant in modern society. As it exists, many flaws remain embedded in governments because few are permitted to attempt repairs.

The state of affairs surrounding political environments in the world today causes societies to evolve rather slowly because they do not utilize the benefits of a decentralized and open model. In fact, most, if not all, models currently used rely on centralized authorities. This is prominent in the dominant forms of republics that many people call democracies. Centralized governance leads to difficult dilemmas like corporate lobbying where government decisions are influenced by essentially whoever has enough money or resources available to them. If lobbying gets banned in response, nobody may use that avenue for having a voice in their government, which is restricting. Otherwise, it remains legal and many are outspoken by those more well endowed with support, which is inhibiting. In a more decentralized model, this problem would be broken down quite a bit from the start by requiring lobbyists to operate across many more localized institutions to achieve the same effect it can have today. It will remain a problem as long as people continue to pursue the cathedral model of nation development in widespread use. Aside from its exploitation of centralization, lobbying is a counterproductive practice because it does not necessarily represent the best interest of the people which is all the government need be concerned with. It is very similar to the lack of quality that is characteristic of proprietary works whose proprietor’s intention for the works may be bought. This violates

the trust of those who license the works which presents another potential obstacle the government must avoid, lest it find itself the unfortunate licensee in such an arrangement with its solutions providers. When government truly embraces technology, it would behoove them to take the benefits of FLOSS philosophies and apply them in as many areas as possible.

As government submits more and more to software for the accomplishment of its basic functions, the use of FLOSS can assist in a number of ways. As discussed earlier, it tends to be highly reliable which is absolutely necessary for the scale a local or federal government could deploy it at. The quality it could be expected to produce would never diminish due to support from invested and interested communities that could stimulate continued growth through collaboration on the creation of more great things like the Internet. The battle-hardened security advantages FLOSS has would meet the necessity for privacy that many government applications would demand. For instance, these may include the storage or transmission of sensitive data such as citizenship statuses, social security numbers, criminal records, etc. Above all else, even if all of these likely outcomes were omitted, FLOSS integration would still be worth the effort for the transparency it would guarantee. Citizens would know exactly how their government handles their information, where their money goes, and what their administration is doing for them. These solutions provided to the government would be developed and maintained by the people. As a great president once suggested, “It is rather for us to be here dedicated to the great task remaining before us...that government of the people, by the people, for the people, shall not perish from the earth” (Abraham Lincoln).

In the spirit of exceeding Mr. Lincoln's resolution, perhaps that is not the only extent to which FLOSS philosophies and methodologies can be used to improve the world's national development standard. Adopting these methodologies could have much more profound effects that change the way governments approach problems altogether. For instance, the quality-producing aspect of the FLOSS model essentially makes government operation debuggable. Quicker eradication of the inefficiencies in government and society leaves only reliable and massively deployable governing architecture suitable for standardization. Decentralized "nodes" employing such a standard would enjoy high security and extreme robustness. On a significant side note, it is much easier to leverage terror against a centralized government than against many collaborative, localized ones. The most fundamental reason, though, is again transparency. Since permissible open scrutiny is required by FLOSS philosophies, drastically less opportunity for corruption and bias from personal interest would be created, and the global focus could shift much more from the logistics of government, to the real issues that need solutions worldwide like poverty, disease, and education. The problems that matter most are ones that are shared by the entire human race. If an open source model for government were standardized and voluntarily adopted, the resulting cooperation and collaboration would change the course of history, one we could have more faith in actually seeing due to the self-perpetuating and sustainable nature of bazaar development.

Seeing these trends of autonomy and decentralization embodied by FLOSS improve the quality of participants' work and digital connections suggests that similar practices may also be used to improve our social connections. There has been a great

effort to promote openness on the Internet, and the evolution of the Internet has only benefitted from it. It now inspires more vibrant creativity across all kinds of media and encourages productivity and innovation at exponential levels. It has become a real world example of how the application of FLOSS philosophies and bazaar development can succeed in contexts other than software production, which has served as a good starting point. Effort should be focused on creating communities like those found around FLOSS projects so that people can begin to reject opinion based thinking for free-data backed, fact based thinking. FLOSS philosophies and methodologies developed autonomously around modern operating systems and revolutionary communication networks, that is, systems that: use abstraction and collaboration to solve problems, share resources for the benefit of all, and facilitate sustainability and transparency. Imagine the possibilities of a world that operated like such a system.

Works Cited

- "WorldWideWeb: Proposal for a HyperText Project." Message to Tim Berners-Lee. 12 Nov. 1990. E-mail. http://www.w3.org/Proposal.html
- Bollinger, Terry. United States. Department of Defense.*Use of Free and Open-Source Software (FOSS) in the U.S. Department of Defense*. Washington D.C.: MITRE Corporation, 2003. Print.
- Gates, Bill. "The Internet Tidal Wave." Letter to Executive and Direct Reports. 26 May 1995. MS. Microsoft, Redmond, Washington.
- "Global Stats." *StatCounter*. N.p., 28 Feb 2013. Web. 28 Feb 2013. <http://gs.statcounter.com/>.
- Hartung, Adam. "How Facebook Beat MySpace." *Forbes*. N.p., 14 Jan. 2011. Web. 21 Mar. 2013. <http://www.forbes.com/sites/adamhartung/2011/01/14/why-facebook-beat-myspace/>.
- Hauben, Ronda. "The Internet: On Its International Origins and Collaborative Vision (A Work In Progress)." *Amateur Computerist* 12.2 (2004): n. pag. Web. 19 Mar. 2013.
- Himanen, Pekka. *The Hacker Ethic and the Spirit of the Information Age*. Random House, 2001. Print.
- "History of the OSI." *Open Source Initiative*. Web. 28 Feb 2013. <http://opensource.org/history>.
- Hoskins, William. "To All Licensees, Distributors of Any Version of BSD." *UC Berkeley*. N.p., 22 Jul 1999. Web. 28 Feb 2013.

<ftp://ftp.cs.berkeley.edu/pub/4bsd/README.Impt.License.Change>.

- Howard, Philip N., Aiden Duffy, Deen Freelon, Muzammil Hussain, Will Mari, and Marwa Mazaid. *Opening Closed Regimes*. Working paper no. 2011.1. N.p.: University of Washington, n.d. Web. 21 Mar 2013. <http://pitpi.org/index.php/2011/09/11/opening-closed-regimes-what-was-the-role-of-social-media-during-the-arab-spring/>.
- "Internet Usage Statistics." *Internet World Stats*. Miniwatts Marketing Group, Jan. 2008. Web. 20 Mar. 2013. <http://www.internetworldstats.com/stats.htm>.
- Lyons, Daniel. "Linux Rules Supercomputers." *Forbes*. 15 Mar 2005: Web. 28 Feb. 2013.
- Moore, J. T. S., dir. *Revolution OS*. Perf. Stallman Richard, Torvalds Linus, Raymond Eric, and Perens Bruce. 2001. Film. 28 Feb 2013.
- Perens, Bruce. *Open Sources: Voices from the Open Source Revolution*. 1. O'Reilly Media, 1999. Print.
- Raymond, E. S. *The Cathedral and the Bazaar*. O'Reilly Media, 1999. Print.
- Richardson, Marjorie. "Interview: Linus Torvalds." *Linux Journal*. 1 Nov 1999. Print. <http://www.linuxjournal.com/article/3655>.
- Ritchie, Dennis. "The Evolution of the Unix Time-sharing System." *AT&T Bell Laboratories Technical Journal*. 63.6 (1984): 1577-93. Print.
- Salus, Peter H. *A Quarter Century of Unix*. Addison-Wesley, 1994. Print.
- "September 2012 Web Server Survey." *Netcraft*. N.p., 10 Sep 2012. Web. 28 Feb 2013. <http://news.netcraft.com/archives/2012/09/10/september-2012-web-

server-survey.html>.

- St. Laurent, Andrew M. *Understanding Open Source and Free Software Licensing*. 1. Sebastopol: O'Reilly Media, 2004. Print.
- Stallings, William. "Operating Systems: Internals and Design Principles" 5th ed, page 91. Pearson Education, Inc. 2005.
- Stallman, Richard. *Free Software Free Society, Selected Essays Of Richard Stallman*. 2. Boston: Free Software Foundation, 2010. Print.
- Stallman, Richard. "GNU Manifesto." *Dr. Dobb's Journal of Software Tools*. 10.3 (1985): 30. Print.
- Stallman, Richard. "The Free Software Definition." *GNU Project*. Free Software Foundation, 20 Feb 2012. Web. 28 Feb 2013. <http://www.gnu.org/philosophy/free-sw.html>.
- Stallman, Richard. "Why Open Source misses the point of Free Software." *GNU Project*. Free Software Foundation. Web. 28 Feb 2013. <http://www.gnu.org/philosophy/open-source-misses-the-point.html>.
- Tanenbaum, A. S. *Operating systems: Design and implementation*. 3. Prentice Hall, 2011. Print.
- Torvalds, Linus. "Linux 0.12 Release Notes." *The Linux Kernel Archives*. N.p., n.d. Web. 28 Feb 2013. <http://www.kernel.org/pub/linux/kernel/Historic/old-versions/RELNOTES-0.12>.
- United States. National Institute of Standards and Technology. US Department of

Commerce. *Federal Information Processing Standards Publication*. N.p.: NIST, 2001. Ser. 197. Web. 19 Mar. 2013.

- United States. National Institute of Standards and Technology. US Department of Commerce. *Guide to General Server Security*. By Karen Scarfone, Wayne Jansen, and Miles Tracy. Gaithersburg: NIST, 2008. Web. 19 Mar. 2013.
- Whittaker, Zack. "Facebook Hits 1 Billion Active User Milestone." *CNET*. N.p., 4 Oct. 2012. Web. 21 Mar. 2013. <http://news.cnet.com/8301-1023_3-57525797-93/facebook-hits-1-billion-active-user-milestone/>.
- World Wide Web Consortium. MIT. *W3C Patent Policy*. *W3C*. N.p., 5 Feb. 2004. Web. 20 Mar. 2013. <http://www.w3.org/Consortium/Patent-Policy-20040205>.
- Wennergren, avid. United States. Department of Defense.*Clarifying Guidance Regarding Open Source Software (OSS)*. Washington D.C.: 2009. Print.
- Williams, Sam. *Free as in Freedom*. California: O'Reilly, 2002. Print.