NONPROFIT ORGANIZATIONS COULD RUN ON OPEN SOURCE SOFTWARE

Presented by

Sergio Reyes

to

Professor Dawn Zapata

Faculty Advisor

in partial fulfillment of the

requirements for the degree of

Bachelor of Science in Management

Cambridge College

Cambridge, Massachusetts

December 2014

Acknowledgments

Special thanks go to my wife Susana for supporting my efforts to reenter the world of formal education after a whole life of work and self-education. Her support was not only psychological but also practical, from encouragement to coming to pick me up at night, in heat or cold.

NONPROFIT ORGANIZATIONS COULD RUN ON OPEN SOURCE SOFTWARE

Sergio Reyes

December 2014

Abstract

This paper addresses the disconnection between the efforts of the open source software (OSS) community and the nonprofit community. In spite of the quality software produced by the open source software community the nonprofit world doesn't seem to realize that they can use the results of those efforts to run their organizations rather than being married to proprietary, many times expensive, software.

An effort is made here to understand the behavior of nonprofit organizations to be hesitant to explore the open source software alternatives. At the same time, a serious look at the history of the OSS movement is paid and to some efforts, both successful and failed, to incorporate OSS in the corporate, public sector or nonprofit world.

In the end the paper provides a practical, managed approach to exploring and deploying OSS in the nonprofit enterprise. We pay special attention not only to the technical aspects of a project like this, but also to the careful planning of the conversion project, and the stakeholders that are key to the success of OSS adoption.

This research concentrated on the available literature, much of it in digital format, to learn from the experience of organizations that have tried to incorporate OSS in their operations. Fortunately, there are a few key studies from where we could draw important information to inform this work.

This paper, after reviewing the history of the open source software movement and some experiences in deployment, proceeds to provide a systematic approach to the assessment and implementation of open source software. Interestingly, by attempting to implement open source and free open source software, the nonprofit world can also contribute to improving and giving strength to the open source software movement.

Table of Contents

Overview

Statement of Problem

Small businesses and non-profit organizations of all sizes seem reticent to use either open source or free software in spite of the advanced development of these computer tools. At the management level there is little knowledge about the existence of this alternative to commercial, for-profit software. At the non-profit management level there is also little knowledge about the philosophy and purpose of these professional efforts carried out throughout the world to produce quality software. This is all compounded by the deliberate efforts made by large software companies to keep the nonprofit sector under their control.

Nonprofit organizations have come a long way since the days when technology was conceived as a luxury or tools that only large business had the capability of acquiring and using. Many small nonprofit organizations used to be the recipients of hand-me-down equipment that commercial corporations basically unloaded on them as they replaced equipment on a fixed schedule, to keep up with changes and new developments. This initial dependency on hand-me-downs could have determined the predominance of use of software that was predominant in the market. In other words, the donated computers came loaded mostly with Windows and Microsoft production software. At the administrative level, Microsoft Windows was the dominant operating system, followed by Apple Computers Mac OS. The latter with greater demand by graphic designers, movie makers and sound systems.

While large corporations appeared to be charitable and supportive of smaller, less affluent nonprofits, they in fact were doing no favors. In fact, they were unloading their used equipment on organizations that played the role of recycling dumps. However, this

would have been an opportunity for nonprofits to strip these computers from their commercial software and install free open source software. Why didn't they?

When nonprofits begin to budget for computers and technology and realize that these are tools of strategic importance, they continue to be the subject of market influence as most manufacturers bundle their systems with Microsoft Windows as part of it. Such is the case of Dell Computers hardware and Hewlett Packard systems. Furthermore, these manufacturers are quick to offer “deals” to their customers to obtain reduced price software such as the Microsoft Office suite.

Microsoft is making a special efforts to maintain nonprofit dependency on their software. To this effect it created a special division that handles licensing for charitable nonprofit organizations. “With the Microsoft Open License for Charities program, eligible nonprofit organizations can acquire multiple software licenses—rather than multiple software packages—at reduced prices. Your organization needs to purchase only one complete software package license and enough licenses to cover the remaining number of computers.” (Microsoft Volume Licensing Reference Guide, 2014, p. 26) The reader should note that the key element on this definition is that their software is offered “at reduced prices” for nonprofit organizations that fall within their “charitable” guideline. One could ask, “Is this real charity, or is it a strategy to keep nonprofit organizations under their market control?”

Microsoft has also partnered with an international nonprofit/NGO serving organization called TechSoup that receives software and hardware donations for distribution to nonprofit organizations for a fee. This is how this organization defines itself. “TechSoup is a 501(c)(3) nonprofit with a clear focus: connecting your nonprofit, charity, or public library with technology products and solutions, plus the learning resources you need to make informed decisions about technology and operate at your full

potential." (www.techsoup.org, 2014, para. 2) . There is no doubt that this organization has contributed to many nonprofits that otherwise would have not been able to afford commercial licenses. However, their emphasis is not in providing non-commercial alternatives to their member organizations. TechSoup makes little effort to promote open source software, even though they are located in a prominent position within the radar of nonprofits looking for technological assistance.

Why are they not promoting also open source software as an alternative source of quality software?

Teaching institutions, at all levels, but in particular at the college level, are also complacent with the predominant software giants, catering in their courses to instruct students in the use of commercial software, completely neglecting to even mention the existence of similar high quality open source software. Many times this happens not necessarily because there is a conspiracy to deny the existence of open source software, but rather because the professors themselves are ignorant about this alternative. In that sense, it becomes necessary to educate the educator on this matter. The problem, naturally becomes, "Who will do this?"

Another problem in adopting OSS in the nonprofit world is peer pressure. At the local levels, nonprofit organizations working in similar areas of service pretty much know each other. Some organizations in trying to justify their lack of interest or their fear to try free as in freedom, and free as in no money, software, cite the fact of "compatibility" with other organizations, with the government, with their funders, and the world.

Finally, the prolific nature of worldwide development of OSS, has, in a sense, also become a part of the problem. There is too much to choose from. We will explore

this situation as a positive situation further in this paper. At this point though this is seen as a problem in contrast to the two software giants currently in the market.

Yet, another problem commonly cited by nonprofit organizations is that open source software doesn't provide technical support. Interestingly, neither does Microsoft, unless of course the customer decides to pay for that service.

Historical Overview

The open source software and free software movements don't have such a long history. Yet, as such, it has reached a level of development that should be respected since it has demonstrated that cooperation without coercion can work efficiently. Carver (2005) explains the beginnings of this movement as follows:

> The free software movement traces its beginning to a jammed printer. In the 1970s, while Richard Stallman was working as a programmer at MIT's Artificial Intelligence (AI) lab, the environment surrounding software development was quite different from today. Prior to the appearance of the personal computer in the 1980s, large mainframes with dumb terminals were the norm. In this hardware-driven world, software was an afterthought and was often provided with human-readable source code at no additional cost, with the purchase of the machines. Software was rarely sold separately. In this environment, Stallman was free to solve a problem the lab faced with sharing a centralized printer—paper jams. With access to the printer's source code, Stallman was able to improvise a solution by modifying the printer software to send everyone a message any time the printer jammed. Anyone who was hoping to receive a printout would then know to go fix the problem. (p. 444)

The story goes on to describe the obstacles confronted by this programmer when trying to apply his code to a Xerox printer, with proprietary drivers code. The idea of free software, free access to the code that makes machines turn, took more than 10 years to become a movement.

> On September 27, 1983, Richard M. Stallman began a software revolution with a post to the Usenet newsgroup, net.unix-wizards. He announced his plan to write a complete software system called GNU that would be compatible with the UNIX computer operating systems in wide use at the time. At the time, Stallman could not have known that the engine of his revolution was not going to be the free software that he and others would write, but a free software license that he would

> develop to implement his vision, the GNU General Public License (GPL). (Carver, 2005, p. 443)

Naturally, the power of a single individual would have never been enough to develop a project of this magnitude. All along Stallman was supported by other like-minded technology workers, mostly programmers, the geeks of yesteryears, who found pride on being called and calling themselves "Hackers." Their philosophy was preserved in the GNU Manifesto, written by Stallman in 1985, which in part states:

> GNU, which stands for Gnu's Not Unix, is the name for the complete Unix-compatible software system which I am writing so that I can give it away free to everyone who can use it. Several other volunteers are helping me. Contributions of time, money, programs and equipment are greatly needed.
>
> So far we have an Emacs text editor with Lisp for writing editor commands, a source level debugger, a yacc-compatible parser generator, a linker, and around 35 utilities. A shell (command interpreter) is nearly completed. A new portable optimizing C compiler has compiled itself and may be released this year. An initial kernel exists but many more features are needed to emulate Unix. When the kernel and compiler are finished, it will be possible to distribute a GNU system suitable for program development. We will use TeX as our text formatter, but an nroff is being worked on. We will use the free, portable X Window System as well. After this we will add a portable Common Lisp, an Empire game, a spreadsheet, and hundreds of other things, plus online documentation. We hope to supply, eventually, everything useful that normally comes with a Unix system, and more.
>
> GNU will be able to run Unix programs, but will not be identical to Unix. We will make all improvements that are convenient, based on our experience with other operating systems. In particular, we plan to have longer file names, file version numbers, a crashproof file system, file name completion perhaps, terminal-independent display support, and perhaps eventually a Lisp-based window system through which several Lisp programs and ordinary Unix programs can share a screen. Both C and Lisp will be available as system programming languages. We will try to support UUCP, MIT Chaosnet, and Internet protocols for communication.
>
> GNU is aimed initially at machines in the 68000/16000 class with virtual memory, because they are the easiest machines to make it run on. The extra effort to make it run on smaller machines will be left to someone who wants to use it on them. (www.gnu.org, 2014, para. 4-7)

While Stallman created a “Unix-like” operating system, he made sure that he would not copy the source code for Unix. Moddy (2002) cites Stallman saying: “I certainly never looked at the source code of Unix," Stallman says. “Never. I once accidentally saw a file, and when I realized it was part of Unix source code, I stopped looking at it" The reason was simple: The source code “was a trade secret, and I didn’t want to be accused of stealing that trade secret," he says “I condemn trade secrecy, I think it’s an immoral practice, but for the project to succeed, I had to work within the immoral laws that existed." (p. 21)

In brief, Stallman re-created the structure of Unix, its command system, its way of processing data, without copying the Unix source code. GNU looked and acted like Unix but it was not Unix.

Eventually, Stallman quit his job at MIT in fear that his body of work would be appropriated by his employer, thereby destroying his claim to write and develop free software. The word “free”, must be emphasized, refers to freedom, not necessarily to free, without cost.

Many years later, Stallman and his dispersed team of hackers felt that GNU was almost ready, all they needed now was to build the “kernel”. A kernel can be defined as a “computer program that manages input/output requests from software, and translates them into data processing instructions for the central processing unit and other electronic components of a computer.” (Wikipedia, 2014, para. 1) This is an essential part of an operating system. Without this kernel the whole structure built by Stallman and others would simply not work. Well, in fact that structure did work, but on top of proprietary systems and that was not Stallman's ultimate goal.

By 1990, two young Danish hackers, Linus Torvalds and Lars Wirzenius, already aware of Stallman's GNU project, decided mostly under the impetus of Linus that they were not willing to wait for Stallman's kernel –called the GNU Hurd, which Stallman planned to build after he finished the operating system. Linus decided then to write his own kernel. However, it was not Wirzenius who could support the programming feat Linus undertook, but rather another more experienced programmer on a teaching operating system called Minix. His name was Bruce Evans (Moddy, 2002, p. 38)

Once the operating system and the kernel were in place the development of application software that would run on top of it simply took off. What is important to note here is that all developments operated under the same community spirit and under the GNU licensing structure, which has now become legally accepted and recognized in courts of law around the world.

One of the outcomes of principles applied to practice in the world of OSS was a type of license that is currently accepted by the courts of the world, including capitalist countries. “The most popular open source license is called the GNU Public License (or GPL). The GPL stipulates not only that the source code needs to be available, but also that the program can be modified and re-distributed, as long as that re-distributed program is also governed by the GPL.” (Murrain, 2007, p.7) Likewise, this license also indicates that if portions of it are used combined with closed source, proprietary code, then the owner of the code must make his/her code also open. Otherwise, it is not acceptable to use open source code.

Currently there are hundreds of “flavors” of open source operating systems, simply referred to as Linux, developed by different communities. Each one of these distributions, simply called “distros” on Linux circle, concentrates on different aspects of service to their potential constituency. The Android Operating System that runs on cell phones and

tablets is a Linux distribution. Other distributions concentrate on running in minimal space. Most of the video surveillance systems run also on Linux.

Some large efforts, such as Red Hat Linux, developed mostly in the United States have concentrated on the business community and eventually they transformed that effort into a successful capitalist organization, making profit out of open source. However, in order to continue using and even expanding their Linux distro, they must keep their source code open. If you take a look at their website, www.redhat.com, you will see that they have covered all aspects of business service including training and technical support, naturally for a fee.

Another distribution of international note is Ubuntu Linux. They describe their purpose as follows: “Linux was already established as an enterprise server platform in 2004, but free software was not a part of everyday life for most computer users. That's why Mark Shuttleworth gathered a small team of developers from one of the most established Linux projects – Debian – and set out to create an easy-to-use Linux desktop: Ubuntu.

The vision for Ubuntu is part social and part economic: free software, available to everybody on the same terms, and funded through a portfolio of services provided by Canonical.” Mr. Shuttleworth (born 1973) is a South African multimillionaire, famous for being the first private traveler to space on the Russian spaceship Soyuz in 2002. In fact, Ubuntu has become a popular distro because of its increasing ease of setup and and use, and unlike Red Hat, the OS continues to be free.

A final piece of information to understand some of the public debate going on within the OSS and FOSS communities refers to the different degrees of the “purity” of the concept and practice of free software. In fact, many communities have become full fledge

corporations for profit, as we have seen with the example of Red Hat. Yet, even those companies subscribe to the concept of open source.

This is corroborated by Towle, McFarland, and Keppler (2004) when they explain: "Though they share a number of goals, the missions of the free software camp represented by Richard Stallman and the Free Software Foundation, and the open source camp represented by OSI (Open Source Initiative), Linus Torvalds, and the various commercial heavyweights who have thrown their support behind Linux, are different. The Free Software Foundation remains antagonistic toward all forms of proprietary or restricted-source software, while the OSI accommodates certain combined open source and proprietary software development through less restrictive open source license terms." (p. 4)

Purpose

Nonprofits just like open source communities are a contradiction within a capitalist society. Nonprofits in the United States have been given a special tax free status in recognition for their services to the system. In a sense, nonprofits deal with problems created by the same capitalist system that considers them a charity. Nonprofits are an important part of the capitalist system also in terms of their contribution to the economy and social stability. Nonprofits have gone from groups of volunteers to enterprise systems that operate efficiently and providing necessary services for the functioning of society. Open source communities also begun as small groups of volunteers to arrive to communities that have the capability of hiring full time programmers to move projects forward efficiently.

This paper provides critical information to nonprofit organizations – which can indeed be useful to any type of business, about taking a serious look at the principles and benefits of adopting OSS. Rather than taking only an ideological approach, we will also

examine practical ways to look for, evaluate, try, and deploy OSS for many areas of business processing.

In particular, we will address different levels of management within nonprofits, as they are the decision makers that can change entrenched current practices in this area. Most nonprofit organizations, regardless of their progressive social service agendas, continue to be vertical, top-down organizations. It is therefore imperative that executives be educated to these technological alternatives.

We also convey the message to our peers in the information technology departments to become themselves familiar with these software developments, if not experts, so they can influence decision makers in nonprofit organizations.

Likewise, we hope to be of use to the line staff who in the end must use the technological tools that they are provided to carry out their daily tasks. This is perhaps one of the most difficult tasks, as the public at large is at the mercy of daily bombardment of marketing propaganda from the large for-profit software developers.

Finally, we want to make a case for ideological match between the OSS community and the nonprofit corporate sector. The mission of nonprofits is to provide services to the community without making a profit in the process. The business of nonprofits is different from the money-making machines that will not move a finger unless that means generating profits, and not just profits but increasing profits. The OSS movement, from its inception has understood that knowledge should and can be garnished for the good of humanity, not just for the enrichment of the few.

Methodology

Traditional literature research on the subject related to the use, or better the limited use of open source software by nonprofits, doesn't come up with too much material. The research of scholarly papers or books on the Internet or databases, doesn't return much on the subject either. There are more material related to certain very specific adoptions by different industries, but not necessarily the nonprofit world.

The always frowned upon in scholar circles open, global Internet searches point to a limited number of efforts to understand the seemingly lack of intersection between these two areas of the economy, free and open source producing communities and nonprofit organizations. That open Internet search pointed to some classical works such as the one produced by the Non Profit Open Source Initiative in 2007, “Choosing and Using Free and Open Source Software: A Primer for Nonprofits”, by their Executive Director, Michelle Murrain.

Open Internet search also pointed to comments about another important work, “Adopting Open Source Software, A Practical Guide” by Brian Fitzgerald, et al., published by MIT Press in 2011. This important analysis concentrates mostly on adoption attempts in the public sector, including an attempt with limited success by the Commonwealth of Massachusetts in the United States.

Much more research is necessary still in the area of understanding the psychological or practical approach of nonprofit organizations to adopting and using OSS. This will require resources and the support of various nonprofit associations such as the Providers Council in Massachusetts and other similar organizations throughout the country. The present work concentrated in providing information to better understand the OSS community efforts and how to attempt the exploration and adoption of OSS in nonprofits from a management perspective, based on the writer's own management experience and

knowledge and the review of available literature to learn from other attempts, both in the nonprofit, public sector or for-profit worlds.

Literature Review

FREE AS IN FREEDOM

Not without cost

Murrain (2007) provides a clear explanation of the concept that open source software is not without cost, but rather it is determined to be free because the code is not proprietary and can be used and modified by any programmer capable of doing so and for whatever purpose according to his/her needs. The founders of this movement provided the best definition to explain this concept and Murrain quotes them as follows: "The foundational philosophy behind free and open source software as articulated by the Free Software Foundation is that software freedom means:

> The freedom to run the program for any purpose. The freedom to study how the program works, and adapt it to your needs. Access to the source code is a precondition for this. The freedom to redistribute copies so you can help your neighbor. The freedom to improve the program, and release improvements to the public, so that the whole community benefits. Access to the source code is a precondition for this." (p. 17)

Total cost of ownership and nonprofits

Ironically, the same author makes the case to indicate that this freedom usually results in either free use of the software itself or a significantly lower cost than purchasing licenses for the use of commercial software. At the same time, Murrain makes a strong case that nonprofits organizations not only benefit from this, but the concepts should be in line with their own missions. "At the simplest level, software that is without cost (that is "free as in beer") and freely available, without license restrictions or limits, provides clear TCO benefit. From a big picture perspective, though, supporting software that is freely available is actually in line with the missions of many nonprofit organizations – those who are focused on providing resources to make the work and lives of organizations and communities better." (Murrain, 2007, p. 18)

Community involvement

Unlike proprietary software development groups, open source developers are usually accessible. This means that users have opportunities to bring their specific needs to the attention of the software developer community. With active participation of software users in the process of reviewing, improving and expanding the capabilities of the programs in use, very powerful system can be built. “The community ownership model of free and open source software means that involvement in that community can have an impact on the direction that software takes.” (Murrain, 2007, p.19) Murrain dedicates a whole chapter to the issue of support by open source communities. The author then affirms that “nonprofit involvement in free and open source developer communities has direct benefit for nonprofit organizations – NPOs get to be involved in making sure that developers pay attention to their needs.”

A culture of service

The highest ideals of nonprofit organization have to do with the “good for humanity”, without consideration for profit making. This is in direct contrast with the stated goals of capitalist enterprises, whose only reason for exist is the generation of profits for their owners, whether individual or corporate. Likewise, the founders of the open source and free software movement have declared that all human beings have the right to use the intellectual product of programmers. That right should not be limited by the many times extremely high cost of ownership that commercial software companies in their efforts to make money assign to their software.

Walpole, J. (2008) presents that concept asserting that, “Most non-profits find the concept of open source attractive for cultural reasons. The idea that open source software represents value created without the intention of generating profit is a natural fit. If an

organization needs to put time and money into software, they often find it preferable for those resources to support a package that is available to everyone without cost." (para. 34) This hypothesis, however, seems to be disproved by the very limited adoption of open source software by the same nonprofits that should see the intentions of both types of organizations fit neatly.

NO VENDOR LOCK-IN AND SAVINGS

Dependency from vendors

Software giants like Microsoft or Adobe have created a virtual monopoly in many areas of productivity. Nowhere is this more pervasive than in the area of administrative computing. Microsoft Office used to be the de facto standard. Its supremacy has been challenged by new web based mobile technology, such as Google apps. Microsoft is responding with Windows 8 and Office 365. While the technological world is changing drastically Microsoft continues to dominate the office productivity world. Most organizations don't like to experiment with new alternatives once they have found systems that work for them. Therefore, most organizations get into a routine of using the same family of software for generations, becoming dependent from vendors.

Citing a study about the total cost of ownership associated with adopting OSS conducted in England by two major researchers, Lockerbie, Pohjolainen, and Williams (2013) conclude that the major driver for adopting OSS was being freed from vendor lock-in. "Reducing vendor lock-in was the most important factor for respondents when considering OSS adoption and was one of the major issues that the original founders of the open source movement felt that OSS would address (Shaikh & Cornford, 2011). One example where vendor lock-in is particularly problematic is with office productivity suites." (p.7)

Cost savings

A natural driver for many organizations is the reduction of expenses. In the case of nonprofits this savings is important because it could be transferred into supporting improved services in other areas of the organization. Unfortunately, change rarely comes without expenses and in this case the savings might not be immediate but rather realized in a longer period of time. What the literature indicate, though in analyzing OSS experiences is that in most cases this is not the main driver for change. "While cost is not the most significant advantage in adopting OSS, there are cost savings to be made by adopting OSS. A recent report for Transport for London showed savings of up to 80% when moving from proprietary software to OSS (Shaikh & Cornford, 2011). Similar savings were reported from the French National Police Force (FNPF) which reported an annual saving of €2 million after changing from proprietary software to Ubuntu, OpenOffice, and Firefox (Canonical Ltd, 2010). The FNPF report found that a combination of increased costs and maintenance downtime was consuming too many resources. This resulted in a major cost cutting exercise, which saw a significant shift in the organizational adoption of OSS (Canonical Ltd, 2010). In general, cost savings through software tend to be a result of cheaper licensing and support (Shaikh & Conford, 2011)." (Lockerbie, Pohjolainen, & Williams, 2013, p. 8)

Interestingly, proprietary software sells the user licenses for their current release of the software, let say Office 2007, but a new release, say Office 2013, means that in order for the user to get that version he/she must pay yet another fee for the upgrade. This one area where the savings in adopting OSS are more evident. "With proprietary software, users also pay for a fixed version of the software, to which you are entitled to security updates, but you'd need to pay for feature and version upgrades in the future.

With OSS, you are usually always able to use the latest version of the software, with long term support between 3 to 13 years depending on the software. An example of this is that Red Hat provide up to 13 years until a version of its software reaches end-of-life (Red Hat Inc, 2013b). You are still able to upgrade to a newer version of the operating system so that you can benefit from the support lifecycle of that version. Red Hat claims cost savings of up to 34 percent over Microsoft's Server platform, while also giving a competitive support lifecycle to Microsoft (Red Hat Inc 2013c)." (Lockerbie, Pohjolainen, & Williams, 2013, p. 8)

Successful (but overlooked) adoption of OSS

Walpole (2008) reminds us that the work of open source programmers it is already deployed and working in many important areas of the economy. OSS plays a very strong role in developing Internet based applications. Likewise, in the area of customer relations management (CRM).

> "Businesses and non-profits of all sizes from the Fortune 1000 to the federal government have adopted open source software packages for many different purposes. For instance, the Linux operating system and Apache Web servers power much of the Internet. The Firefox Web browser is gaining substantial market share. Content management systems like WordPress, Joomla, Drupal and Plone are widely used by nonprofits, while Constituent Relationship Management systems like SugarCRM and CivicCRM are increasingly viable options. An open source model can result in powerful, secure, useful, industrial-strength software." (para. 11)

INNOVATION, EVEN DISRUPTIVE INNOVATION

The fundamental innovation contributed to humanity by the OSS movement has less to do with the actual products but rather with the concept of how and for what the programming efforts are undertaken. Those who dislike the collectivist concept of OSS try to put it down by indicating that is not an innovation, given that it has not generated a "unique and different" product from others already in existence. However, the innovative

aspect of this intellectual product is that it is transparent to the public. As such, it has many advantages in terms of technological development due to the power of community input. “When code is open, many coders can inspect it, and faults often will be detected more rapidly than when only a handful review it. Furthermore, those who can see the code can suggest improvements and submit code changes. As with all OSS communities, the developers and supporting community members for OSSg2 (*) community members can be recruited on the basis of talent and contribution, unfettered by physical location. This ready supply of programmers ensures innovative ideas can be contributed to the OSSg2 community from both traditional sources and sources previously untapped by traditional software firms. This phenomenon directly attacks innovation risk.” Watson, Boudreau, York, et al., 2008, p. 45) (*) Acronym for second-generation open source.

OSS has presented as serious challenge to commercial, for-profit software developers. Some authors indicate that OSS innovative way to think about and generate software could be a disruptive force for the capitalist software industry. Their conclusion, however, indicate that given that OSS developers are not “competing” with commercial software developers, therefore the impact on sales is not critical. “We showed that open source is a disruptive innovation, but this does not lead deterministically to the failure of the incumbent firm, as the early work by Christensen implies. In fact, studying disruptive innovation is interesting exactly because we don't know ex-ante what will be its market impact.” (Katsamakas, Goergantzas, 2010, p. 227)

Introduction

1) Open source software is a viable option for nonprofits

We have discussed the different good reasons and drivers that support the concept that open source software is a natural option for nonprofit organizations. Ironically, the same reasons that are its strong attributes – free as in freedom, free as in no license price, community development, have kept is as a perceived fringe product. Quietly many for-profit businesses have already adopted it and are using it even to make profits. Open source software is all around us. We carry it daily in our Android phones. If your nonprofit has a website hosted by any large web hosting company, chances are that you are paying, albeit a small fee, for using a portion of a Linux sever, running open source web serving software Apache. Indeed you are paying a low price for web serving because the company providing it is running Linux. Most of the Amazon technological infrastructure is based on Linux, and so it is Google. A large portion of users have installed in their desktops or laptops an alternative Internet browser such as Firefox, which is an open source product.

Thusly, your nonprofit organization is already using a portion of the open source software we discuss here. However, much of this is transparent to you since the operations that support the services you are renting are not run by your own information technology or technological services department, or for smaller nonprofits, your technology support staff or consultants.

Now, you could make a decision to move forward with adopting open source also for your administrative desktop or mobile needs. That would mean replacing Windows for one of the many Linux distros and with it also getting rid of Microsoft Office. It sounds radical and surely many readers already cringe at the idea of dealing with

something different and unknown. A learning nonprofit organization should be open to new initiatives and not look down upon concepts such as technology for the good of humanity. Yet, none of the good reasons nor intentions are in itself a guarantee that changing from proprietary software to open source software will be a smooth and successful process. Nonprofits need to make sure that a thorough analysis and carefully planned process of adoption is in place. Otherwise, it could be a disruptive process that could negatively impact your main operations. Nonetheless, given that all precautions are taken, open source software is definitely a viable option for your nonprofit organization.

a) Learning from other adoption experiences

Even though no large amount of literature about nonprofit adoption of OSS, there are some that are useful for this purpose. One of the important primers to read and analyze is "Choosing and Using Free and Open Source Software: A Primer for Nonprofits", by Michelle Murrain, published by the Nonprofit Open Source Initiative (NOSI). It is unfortunately a 7-year old report, but it is still current in terms of principles, although much has changed in that period. Unfortunately, NOSI appears to have stopped functioning in 2012. No proper death certificate can be found. Their report is still available and it contains invaluable information though.

Another important report for larger, public sector attempts around the world is also appropriate for our subject matter. It is the work of five authors title "Adopting Open Source Software", by Brian Fitzgerald, et al, published by MIT Press in 2011. This book examines attempts to adopt OSS in the U.S., Spain, and Italy. Notably, this book examines a nearly 5-year attempt at the Commonwealth of Massachusetts to transform its software infrastructure to open standard file formats and open source software.

The lessons derived from all of these initiatives, but in particular the State of Massachusetts are significant. We will discuss them here in a summary format to keep as a precedent to take into account the positives and avoid negatives in our own implementation.

a1) Mandate and Executive initiative and buy-in. "In 2003, the Commonwealth of Massachusetts launched a series of new initiatives that sought to foster the use of open standards and OSS. The centerpiece was an open standards policy that required all government-owned IT software to rely on open standards. As a result, Massachusetts became the first U.S. state to adopt an open standards policy." (Fitzgerald, et al., 2011, p. 85)

It should be noted that in 2003 the Governor was the Republican Mitt Romney. It was under his administration that the Commonwealth's IT Department started a radical process of changes in computing in the state. The change to mandate the use of open standards for file documentation meant that proprietary file formats such as Microsoft Word's .doc, Excel's .xls or PowerPoint .ppts, were not acceptable formats. It took about 5 years to come up with a conclusive mandate which took the format or a manual. "The final policy was an enterprise technical reference manual, which provided a framework for the standards, specifications, and technologies that must be incorporated into prospective IT investments (Commonwealth of Massachusetts 2008)." (Fitzgerald, B. et al., 2011, p. 89). By 2008, a Democrat, Governor Deval Patrick had taken office. Microsoft responded by incorporating open standard formats in their software output.

The fact that this project was supported by the Governor himself and those under his direct command, allowed IT to move in all areas of the administration with the

authority to make changes. This was definitely a positive attribute of this process and one that should be emulated.

a2) IT cannot be left alone in the process. Engaging stakeholders.

The case of the Commonwealth of Massachusetts illustrated the fact that the department leading the charge and ultimately responsible for the implementation of open standards file formats and open source software was seen as an imposed measure. The Commonwealth failed to include other parts of the organization, in order to make it more of a collective and democratic process. At the same time, change was delivered down the chain of command and user as orders in the form of a Users Manual to be adhered to. Likewise, the measures taken didn't consider the impact of the project on important external and internal stakeholders. “The state government faced considerable criticism for its decision from the Massachusetts Software Council, the legislature, and advocacy groups.” (Fitzgerald, et al., 2011, p. 94)

In fact, the whole project was put to an end when one of those advocacy groups actions, State workers with disabilities. “Before Massachusetts mandated ODF, disabled workers and their representatives voiced their concerns to the ITD. According to the president of the Disability Policy Consortium, John Winske, these concerns met with no response from ITD. This turned out to be a mistake for the ITD. The representatives of disabled workers took their concerns to the press, where they received sympathetic attention.” (Fitzgerald, et al., 2011, p. 98)

a3) Not underestimating compatibility issues.

It is important to recognize that there isn't 100% compatibility between OSS and proprietary software, such as the Microsoft Office suite or others. OSS developers have not copied nor mimicked MS Office. This is a fact that Massachusetts' ITD

underestimated. "Massachusetts initially believed that the adoption of ODF and the shift from Microsoft Open to OpenOffice would not be very complex. However, once it began the transition, Massachusetts was forced to switch from using and open source office suite back to Microsoft Office with plug-ins. This was done to accommodate the community of people with disabilities. However, this change also proved problematic, because plug-in solutions did not yet exist and had to be developed. The resulting technical issues were unforeseen and prolonged the process of implementing ODF." (Fitzgerald, et al., 2011, p. 98)

At this point the impetus for effecting change must have been weakened by resistance and opposition to the project. It is likely that if the affected community would have been invested in this change, rather than resorting to developing plug-ins for Microsoft Office, the OpenOffice development group could have been engaged in producing the elements needed by the community of State workers with disabilities. However, even if this was not the case, hopefully the OpenOffice developing group took note of their deficiency in service to this very important stakeholder.

2) How it can be implemented in your nonprofit

Change is never easy to implement and much less to manage it in such a way that it will not backfire. The experience of the Commonwealth of Massachusetts speaks to many years of vertical initiatives and resources spent in the process of implementing open standard file formats and open source software. In the end, it is possible that all that remained was the adoption of open standard file formats. Nonprofit organizations, large or small, must be extremely careful and organized when attempting to implement this change.

a) Management education and buy-in

A very important sector of stakeholders and change makers is the management structure, starting at the very top. Without the support of managers across different departments in a nonprofit organization, it is likely that the implementation will not be successful.

a.1. The role of executives

It is possible that executives, given that they are fully immersed in the specifics of the mission of the nonprofit organization, are not necessarily familiar with technological trends. In general, they are counting on their information technology departments to provide them and the organizations tools that allow operations to work smoothly.

Besides a general case for open source adoption, it is necessary then to generate a business case for incorporating OSS into an organization. Elements such as total cost of ownership, return on investment, strategic alignment, philosophical alignment, innovation, and encouraging the exercise in practice of a true learning organization, are all elements that should be part of the reasoning used to connect with the top executives and directors that rule the organization.

Arranging a series of thought provoking, well designed presentations to educate executives about the OSS movement, its developments, achievements and future possibilities would be necessary to get their attention and to start assuming the challenge of change. Internal advocates for OSS will play a central role in this process. However, engaging external experts and hopefully principals from organizations where the switchover from proprietary software to OSS was achieved successfully would go a long way in gaining support from the executive group.

a.2. The role of IT departments or staff

No assumption should be made that your Information Technology Department is composed of professionals who are OSS knowledgeable. It is more likely that while these professionals are aware of the existence of OSS, they are not experts on it. These professionals, however, are a key stakeholder in this process of change and in order to support change, they must first be ready for it, both technically and psychologically because they will play a central role in the implementation of new software running on either new or existing hardware.

The first group to be directly affected by shifting from proprietary software to open source will be the technical support, or information technology group. Without their support no change can even be thought of. A key contributor to this process is the director of that department or function, whose leadership will be central to providing the necessary resources first to his team and then to the entire organization.

The IT department, regardless of whether there is a plan to switch to OSS or not, has to be a true learning organization, not only having a general knowledge of technological trends but using and experimenting with different systems and software. IT professionals trained in different systems and software are in a unique position to be able to understand the differences, the peculiarities, the challenges, and the benefits of OSS. This means that it is extremely important that all IT personnel be trained and comfortable in practice with OSS before they can act as advocates and trainers of the new software.

Management will need to complete an honest assessment of the skill sets of their IT debarments with respect with OSS. In case of deficits in knowledge, training and the support of expert consultants will be necessary.

b) An adoption plan

A carefully thought out plan should be put in place once the decision to explore adoption of OSS is made. A formal OSS Implementation Team should be created with a clear charter and mandate from the executive. A reasonable amount of time should be allocated to this process, depending on the size of the organization from 6 months to a maximum of 2 years. A large organization like the Commonwealth of Massachusetts took a bout 5 years in the process, with clear mandate and authority from the Governor, and yet the final result was an incomplete process of adoption.

b.1. Assessing current technology and its use

The IT department should provide a complete technical inventory of equipment, including hardware and software specifications. Staff using the equipment should complete a questionnaire indicating for what they use the software installed in their systems. This inventory will provide a base to assess what OSS could replace the functions performed by the currently installed proprietary software.

b.2. Stakeholders analysis and risk analysis

A thorough stakeholders analysis is very important to understand how change will impact different users, what their reactions could be, what their need would be to become fully functional with the new software packages that will be provided to them for their use. The stakeholders analysis will also be used to create a meaningful communications plan about the conversion project.

The risk analysis will concentrate on anticipating the risks involved in the conversion process and how to avoid or mitigate them.

b.3. Cross interest, cross functional teams

An organizational cross functional, cross departmental team should be created to implement the OSS adoption process. This means engaging the team performers in deep education about OSS. This team will not serve simply as an advisory team, but rather will carry out different tasks needed in the conversion process. A large part of their tasks will involve advocacy of the use of OSS.

b.4. Analyzing alternatives

One of the function of the team will be to analyze different software equivalences and replacements and select the OSS software packages to be adopted. Besides the technical capabilities of the software proposed, it is important to make contact with the community, groups or companies, developing the selected software. This will be particularly important if there are needs presented in the conversion process that are not programmed in the software. It is possible that the same programming community could respond to the request made by your organization.

b.5. Progressive implementation and education of staff

Once a set of OSS packages has been selected, an education and training process of selected staff should be carried out. The groups or departments selected should be given all necessary technical support to carry out their tasks successfully.

b.6. Total adoption

Total adoption should only be attempted once the majority of the organization's staff has been trained and have proven that their jobs can be carried out effectively using OSS. It is critical to monitor and control the entire process of adoption, paying special attention to the difficulties encountered in the performance of important tasks.

Conclusion

Open source software is present throughout the world, many times in an invisible way. In a sense OSS is a well-kept secret by economic systems that prefer the public not to know too much about it. The software production process is as revolutionary as its output, and the output is not only transparent in terms of access to the programming code, but is also efficient in production terms.

Yet, there seems to be a basic mistrust of this software by end users. Ironically, for-profit companies have been offering systems for a fee that work based on OSS. There are examples of this situation throughout this paper.

Nonprofit organizations, also a peculiar kind of enterprise within the capitalist system as they provide services without profit objectives, could be a natural match for OSS. But, even organized efforts such as the Nonprofit Open Source Initiative (NOSI) gave up after many years of trying.

In spite of the skepticism about OSS, there are still many reasons why it is a good thing that should be explored and tried. The benefits of this attempt should not only be seen in the narrow scope of how a nonprofit can directly and immediately adapt and improve its operations using OSS, but rather how in the process of doing so the nonprofit also contributes to enhancing the community effort to produce free, as in freedom, software. In other words, how the efforts combined of the two concepts thriving at the margins of capitalism can contribute to the good of humanity.

References

Brian, F. et al. (2011). *Adopting Open Source Software, A Practical Guide.* Cambridge, MA: MIT Press

Carver, B. (2005). *Share and Share Alike: Understanding and Enforcing Open Source and Free Software Licenses.* University of California, Berkeley Technology Law Journal (pps. 443-481)

Katsamakas, E. and Goergantzas, N. (2010). *Open source disruptive-innovation strategy.* Human Systems Management 29, pps. 217-229.

Lockerbie, S., Pohjolainen, S., & Williams, S. (2013). *Corporate Adoption of Open Source Software*. University of Oulu, Finland

Microsoft Volume Licensing Reference Guide, March 2014

Moddy, G. (2002). *Rebel code, The inside story of Linux and the open source revolution.* New York: Basic Books [Kindle file]

Murrain, M. (2007). *Choosing and using Free and Open Source Software: a primer for nonprofits.* Nonprofit Open Source Initiative (NOSI)

Rubens, P. (2013). *How to Run Your Small Business with Free Open Sources Software.* Retrieved from http://www.cio.com

Techsoup. http://www.techsoup.org/about-us/what-we-do. Retrieved 11/3/2014.

Towle, H., McFarland, C. and Keppler, E. (2004). *Open source issues in business*. Journal of Internet Law. December 2004. pps. 3-11.

Walpole, J. (2008). *Open Source vs. vendor-provided software: comparing them side by side*. Retrieved October 6, 2014 from www.idealware.org

Watson, R., Boudreau, MC., York, P., et al. (2008). *The Business of Open Source.* Communications of the ACM, April 2008. Vol. 51 No. 4

www.gnu.org. https://www.gnu.org/gnu/manifesto.html. Retrieved October 8, 2014.

www.wikipedia.org. http://en.wikipedia.org/wiki/Kernel_(operating_system). Retrieved October 8, 2014

Appendix 1

A Sample Table of Selected Software Equivalences

Function	OSS	Proprietary
Desktop, laptop, server operating system	Ubuntu Linux	Microsoft Windows, Apple OS
Word processing, spreadsheets, presentations, business graphics	LibreOffice	Microsoft Office
Database serving	MySQL	Microsoft SQL
Advanced graphics and image editing	GIMP	Adobe Photoshop
Business graphics, flowcharting	Dia	Microsoft Visio Professional
Project Management	ProjectLibre	Microsoft Project
Web browsing	Firefox	Microsoft Internet Explorer
Sound recording	Audacity	Avid's Pro Tools
Video editing	Kdenlive	Adobe Premiere Pro
Accounting	XIWA	QuickBooks Pro
Computer Aided Design	BRL-CAD	AutoCAD