Community and Commercial Strategies in Open Source Software

Anthony I. Wasserman

Carnegie Mellon University – Silicon Valley
Moffett Field, CA 94035 USA

tonyw@sv.cmu.edu

Keywords: software business, open source software, FOSS, FLOSS

Abstract

This paper describes the recent evolution of business strategies used by companies offering products and services based on free and open source software (FOSS). The primary focus is on companies that develop and release products under an open source license. The paper compares their practices with traditional proprietary software companies and with community-based open source projects, and identifies growing overlaps between the different kinds of software companies. Finally, the paper describes the likely impact of recent technology developments in mobile and cloud computing on open source software and related business.

1 Commercial and FOSS Software: a Brief Overview

The commercial software industry dates to the mid-1960's, when software was first sold independently of the computer on which it ran. The dominant sales model for this enterprise software has been an initial fee that includes a commercial license to use the software. Customers have the option of buying product support, typically at an annual price of 15-20% of the original list price, that entitles them to regular updates and personalized technical support [1]. These software companies also sell professional services, including training, product customization, and product integration ("the full product"). While they may have a network of consultants, distributors, and resellers, changes to the product itself are made by the company, which determines product features, priorities, and release schedules.

Since the emergence of the personal computer industry in the early 1980's, there has also been a commercial market for consumer software, initially available primarily through retail channels, but now more commonly sold online for immediate download or provided as an online service for a subscription fee. End-user license agreements (EULAs) govern the use of such software, typically restricting the installation, use, and redistribution of the software, as well as limiting vendor liability for harm to the consumer or the equipment on which it is used.

In addition to these commercially-licensed products, there has long been software available free of charge, dating back to the earliest days of the computer industry. Universities and research laboratories, both commercial and government-based, frequently provided their software, including source code, to interested parties. Even after the markets for commercial software developed in the mid-1960s, many organizations continued this practice.

In the early 1980s, Richard Stallman created the GNU project at MIT [2], and freely distributed that software to anyone who wanted it, in keeping with his belief that all software should be free to modify and distribute. He subsequently created the Free Software Foundation (FSF), which emerged as a leading advocate for free software. The Open Source Initiative (OSI), created in 1998, produced the Open Source Definition [3] and approves software licenses that are compliant with that

Definition, including those created by the FSF. Many of these OSI-approved licenses anticipated the use of open source software in commercial products and did not include the restriction on contributing modifications back to the project that is found in the FSF licenses. The emergence of open source software disrupted traditional software businesses in the sense that organizations no longer had to rely solely on commercial vendors, contract software developers, and/or their own internal development efforts.

A highly visible example of this new option was IBM's decision to include the highly popular open source Apache HTTP server in their WebSphere Application Server; that decision attracted attention from business journalists [4]. With WebSphere holding a substantial share of the application server market, WebSphere customers, which included many of the world's largest companies, were using open source software in their business-critical applications.

From there, the commercial opportunities for FOSS grew rapidly. Today, one can choose among several types of FOSS software in addition to proprietary software, either installing it locally, or on a hosted server (physical or virtual). FOSS has had a significant impact not only on traditional desktop and server software, but also on mobile computing, embedded, and cloud platforms, making it a significant factor in software adoption and use for the foreseeable future. Many of the leading vendors of proprietary software, including Microsoft, Oracle, and IBM, now offer FOSS alongside their "closed" products, reflecting a change in their software business strategies.

In the remainder of this paper, we shall primarily focus on FOSS, addressing several different approaches to building businesses on FOSS as well as the growth of FOSS. In Section 2, we discuss non-commercial or "community" approaches to FOSS. In Section 3, we discuss the commercial approaches to FOSS, and try to compare them with both community-based and proprietary ("closed source") software. Section 4 is an overview of the various business models used by companies offering FOSS products and services. Finally, we discuss in Section 5 the likely impact of trends in mobile and cloud computing on FOSS.

2 Community-based FOSS

In this section, we review the two major types of non-commercial FOSS projects, which we term independent and foundation-based. We devote particular attention to Linux, since it is one of the most important FOSS projects and has both community and commercial aspects.

2.1 Independent FOSS projects

Hundreds of thousands of FOSS projects have been undertaken, with many of them made widely available in public repositories ("forges"). Since anyone can create, use, or fork a FOSS project, these forges contain projects in widespread use as well as those created primarily for the developer's own use or as an academic exercise. Highly active projects, such as the VLC media player, the Ruby on Rails system, and the FileZilla issue tracking system, are easily identified through visible project information, such as the frequency of updates, the number of followers and milestones, along with counts of downloads, code commits and user recommendations. These projects, as with most projects in these forges, are developed and maintained by a community of volunteers, with a small number of people on each project authorized to commit changes to the main code base.

We shall refer to these projects as "independent" community FOSS projects, in contrast with FOSS community projects that are hosted by a non-profit foundation, each of which provides support for one or more projects, forming a community of developers and users.

2.2 Foundation-based FOSS

"Foundation-based" community FOSS projects differ from independent community FOSS projects through the stewardship and governance of their projects. The Linux Foundation, Free Software Foundation, the Apache Software Foundation, the Mozilla Foundation, the Document Foundation, and the Eclipse Foundation are home to numerous FOSS projects, including the Apache HTTP web server, the Eclipse programming environment, the Subversion version control project, the Firefox web browser, LibreOffice, and others. While there are significant differences among the governance

of these projects, they all share the property that anyone has access to the source code and that anyone can participate as a member of the community around any project as a user, tester, or contributor. Most of these foundations have created their own license governing the use and distribution of the software associated with its projects. These foundations receive funding from corporations and other financial supporters to support the costs of project infrastructure and foundation management. That funding, combined with the screening and review of the various projects, attracts a larger community of developers and users, which, in turn, means that the foundation-based projects are quite stable.

2.3 Linux Distributions

The most widely used foundation-based FOSS software is Linux, originally developed by Linus Torvalds beginning in 1991. Today, development of the Linux kernel is managed through the Linux Foundation. There are hundreds of software distributions ("distros"), some with commercial support, e.g., Ubuntu and SUSE, but many without, e.g., Gentoo and Mint. These distros combine the Linux kernel with a large set of independent and foundation-based FOSS projects, and are typically packaged so that it is a straightforward task to install the Linux kernel with these projects. Beyond that, the distros are combined with updating mechanisms and repositories from which users may select and install additional compatible libraries and applications. In general, the software in these distros has been widely used and tested, with the result that the quality of the software in a distro is very high. Independent studies have found that the quality of this FOSS software is comparable or superior to traditional packaged commercial software [5].

Linux itself is a non-commercial foundation-based project. Many users select a Linux distro for their server and/or desktop and use community-based support mechanisms to address any issues that they may have. In addition, though, some Linux distros are offered with commercial support services, including service level agreements (SLAs) for critical issues and protection against legal claims of intellectual property (IP) infringement.

Here we see a fundamental distinction between non-commercial and commercial FOSS projects. For many individual users, particularly those with technical expertise and those not working on business-critical projects, the non-commercial versions, with their community-based support, are sufficient and offer high quality software at no charge. However, many companies, governments, and other organizations require commercial support similar to that provided by traditional software vendors. For them, Red Hat Enterprise Linux or Ubuntu with paid support from Canonical may be a suitable choice, in contrast to the non-commercial CentOS, openSUSE, or Ubuntu without paid support.

2.4 Support for Community FOSS

A key issue with non-commercial projects is that the software is available "as is", with no *official* support of the type expected by those who pay vendors, e.g., SAP or Oracle, for support of commercial software products. Nonetheless, well-governed projects maintain discussion fora, allowing anyone to post a question or comments, and anyone to respond. In that way, many user issues are easily resolved, sometimes more quickly and easily than with traditional commercial support mechanisms. Such projects also maintain issue tracking systems where users can post bug reports and feature requests. As with commercial software products, though, there is no assurance that the fixes or enhancements will be made.

In addition to the message boards and issue tracking systems, many non-commercial projects maintain IRC chats where it is sometimes possible to get immediate responses to simple issues. Knowedgeable users and project committers often stay connected to an IRC channel as a way to help the community. Finally, there are free third-party sources of support for FOSS, such as Stack Overflow, a question-and-answer site for developers. With these mechanisms, a developer can post an issue or a question on a project's discussion board, interact with project developers on the IRC channel, and post the same issue on a Q&A site. Taken together, many issues are quickly resolved through the volunteer efforts of the community around that FOSS project.

Other widely used and mature open source projects include the Drupal content management system, the WordPress publishing system, and the LibreOffice office suite. These projects were developed independently and non-commercially, following traditional open source practices of community-

based development and support, and remain active, community-based projects. At the same time, though, each has attracted a broad range of related commercial activities, including add-on software, books, and training. This commerce is consistent with the Open Source Definition.

3 Commercially-supported FOSS

There are two distinct categories of commercially-supported FOSS software. The first is where a company offers professional services and products based on community-based software (both independent and foundation-based). For example, Red Hat and Cloudera generate revenue for their businesses through selling subscriptions to the software, offering professional services and add-on products, and/or providing legal assurance services. The second is where a company develops FOSS software with its own engineering team, making that code available to all, and then selling services and add-on products to create a business. JBoss, MySQL, and Zimbra, all of which have been acquired by other companies, all fit into this category, which is more accurately termed "vendor-controlled FOSS" or "vendor-created FOSS".

3.1 Community vs. Vendor-created FOSS

There are some important differences between community-based FOSS projects and vendor-supported FOSS projects. First, the commercial ventures are strongly driven by the business expectations of their paying customers, and by their own goals of generating revenue. Their customers expect to receive product updates on a predictable schedule, as well as to obtain product support from the vendor. That's a strong contrast with the practices of many community-based FOSS projects where a common saying is "the software is ready when it's ready" and where support is provided by the community itself, using online discussion forums as the principal communication mechanism.

Second, the code for a vendor-controlled FOSS product is normally written entirely by the company's employees, unlike the situation with independent and foundation-based FOSS projects where individual core committers lead development of the project, but can accept contributions from others. While many FOSS vendors have developed agreements to accommodate contributions from non-employees, such code contributions to the core product are uncommon. One side effect of this approach is that vendor-controlled FOSS projects rarely make the pre-release source code available. Community-based projects, both independent and foundation-based, by contrast, usually provide continuous access to the source code, including the nightly builds.

Similarly, there are some important differences between FOSS products and proprietary software, the most obvious one being the availability of the source code itself, with the attendant possibility of modifying that code to meet individual requirements. In many ways, though, the most important difference is that customers of proprietary software products must pay an initial license fee for the software while the users of open source software products do not. Also, traditional proprietary software vendors, such as SAP, offer their products only under a commercial license, whereas many commercial FOSS companies offer their products under both an approved open source license and a commercial license. These distinctions lead to different business practices between the two types of companies, especially with respect to sales and marketing [6].

Without access to source code, end users of proprietary products are entirely dependent on vendors to respond to bug reports, enhancement requests, and customization needs for their software. Severe bugs and security issues often result in an unplanned update that can be downloaded from the vendor's servers. However, other requests may or may not be fulfilled, depending on the company's product roadmap and competing requests. Users of FOSS products have the option of modifying the code on their own, though it may be impractical to do so, either because of code complexity or because of difficulties in obtaining vendor support for the modified code.

3.2 Why Vendor-Created FOSS?

Vendor-created FOSS occupies a middle ground between community-based FOSS and proprietary software. As with proprietary software vendors, FOSS vendors offer professional services, including product support, training, and system integration. Since these vendors have developed the code, they

are able to offer service level agreements for product issues, as well as the broad spectrum of phone-based support, message boards, and live chat.

These FOSS vendors can concurrently accommodate the needs of users who want to use the software at no charge and those who want to pay for the support services. These vendors hope that some "free" users will eventually need support services and will thus convert to paying customers. An example of such a transition is a startup software company that intends to include that software in a product that they offer to customers. To save money, they may wait until they are ready to deliver their product to customers before they sign up for commercial support.

MySQL, now owned by Oracle, was an early example (1994) of commercial FOSS. There was a single version of the MySQL source code, available under a dual license model. Anyone could download it and use it indefinitely for no charge. Those customers who wanted to include it in a product that they shipped to their customers were required to purchase a commercial license, thus providing a revenue stream to the company.

Today, though, many FOSS vendors offer more than one version of their product. In that model, there is no charge for the basic "community" project, but the companies have built additional proprietary software components that are offered through a commercial license – not an open source license. This notion is termed "open core" [7,8]. In general, customers of the open core products have access to the source code of the community edition, but not to the open core add-ons. (Jaspersoft is a notable exception to this approach, offering "visible source" to its commercial customers under their commercial license.) Because these open core products are offered under a commercial license, rather than an OSI-approved license, they are not considered to be FOSS.

Nonetheless, these commercial FOSS vendors meet a need in the marketplace. Many organizations are simply seeking the best solution for their needs. For them, source code access is not a factor in their decision-making process, which is based on product functionality and vendor support.

This diversity of needs is recognized by some FOSS vendors. Acquia, for example, was co-founded by the leader of the Drupal project, with the idea that Drupal would remain a community FOSS project, and Acquia would address related commercial opportunities. Potential users can visit the Acquia website and find the community software free for download, just as on the main Drupal site, along with other free software contributed and packaged by Acquia. Acquia's various commercial offerings are also available on the site, thus providing a broad range of options.

In short, there is a demonstrable need for commercial offerings that build upon FOSS. In the next section, we describe a variety of approaches that have been taken toward creating FOSS-related businesses.

3.3 Summary

We have now described five different types of software: independent FOSS, foundation-based FOSS, commercially-supported FOSS, vendor-created FOSS, and proprietary software. The first two of these are community-based, in the sense that they are not provided by a commercial vendor. Commercially-supported FOSS refers to community-based software that is supported by a vendor. The latter two are vendor-based, one available as FOSS and the other being traditional proprietary software ("closed source").

Table 1 shows how these differ with respect to development process, release schedule, support mechanisms, and licensing. Commercially-supported FOSS may be released with either open or fixed dates even though the underlying community software has a more flexible schedule. For example, Ubuntu has a tradition of making two releases of their product each year, at the end of April and the end of October, even though the underlying Linux kernel and other components of the distribution are released more frequently. Note that FOSS released as a project of an open source foundation, such as Mozilla Firefox, always uses the foundation's license, while independent projects have complete freedom in their choice of FOSS license.

	Independent FOSS project	Foundation-based FOSS	Commercially supported FOSS	Vendor-created FOSS	Proprietary software
Development	Community	Community	Community	Single vendor	Single vendor
Release schedule	Open	Open	Open or fixed dates	Product roadmap	Product roadmap
Support	Community	Community	Vendor and community	Vendor and community	Vendor
License	FOSS	Foundation's FOSS license	FOSS and commercial	FOSS and commercial	Commercial

Table 1 – Key differences among different types of software

4 Business Models

There are many different ways by which individuals and companies can create businesses around FOSS [9,10,11,12,13]. There is a long history of individuals, small businesses, and system integrators offering consulting, training, and custom development to companies and organizations, involving both community-based and vendor-based FOSS. Customers pay for time, knowledge, and custom code, not for FOSS. This approach is particularly attractive in locations with relatively little technical expertise, where the service can be provided locally in the local language.

In the remainder of this section, we focus primarily on the approaches taken by commercial FOSS vendors and by companies that support a broader FOSS ecosystem. These approaches are the most common and most important was for businesses to generate revenue from FOSS. Many of them, along with associated sales and marketing strategies, are summarized by Riehle [14].

4.1 Subscription and support services

Subscription services for FOSS are intended to match the update services offered by proprietary software vendors. For a fixed subscription price, users receive such items as downloads, updates, errata, source code, technical whitepapers, solutions, and access to the vendor's knowledgebase. Most customers regularly renew their subscriptions, which provides the vendor with a steady flow of revenue, one that is often more predictable than the traditional proprietary approach of selling software licenses.

In addition, users can sign up for support services at various levels of response. These services can range from "self-service", i.e., similar or identical to the discussion boards available to community users, to multiple levels of premium services similar to those available from proprietary vendors. The premium levels typically include telephone access to support, but also a service level agreement that guarantees response times based on the severity of the identified issue.

4.2 Cloud-based hosting services

Software as a Service (SaaS), originally termed "on-demand software" by Chou [15], uses a cloud-based subscription service, and has emerged as an important model for web-based development and application deployment. Salesforce.com, a proprietary vendor, was among the first companies to build a successful business around this approach. Users don't install any software on their site, but rather pay a monthly or annual fee for access to the hosted software. This approach is becoming much more common, and many business and consumer-oriented applications are now offered as hosted applications.

Many FOSS applications are also now available in this way. Bitnami, for example, has partnered with Amazon Web Services to allow users to deploy more than 35 different FOSS applications, including WordPress, Redmine, and Joomla, in the Amazon cloud. Acquia offers Drupal Gardens, a hosted service for Drupal sites. Other vendors similarly offer FOSS development tools and applications in the cloud. These hosted services are particularly valuable for small and medium businesses, since they reduce their IT equipment and support costs.

In addition, many SaaS companies make extensive use of FOSS for building, deploying, and monitoring their applications, but do not make their source code available. Well-established companies, including Google, Yahoo, and Facebook, have FOSS underpinnings for their hosted services, a trend that is increasing with the greater availability and lowered cost of cloud-based platforms.

The growth of SaaS and cloud-based computing has significant implications for FOSS. Because SaaS applications do not involve distribution of any software, developers of SaaS applications have considerable freedom in their use of FOSS, as many conditions in FOSS licenses only come into effect when the software is distributed to others.

4.3 Dual license model

As noted above, the dual license model gives a vendor the ability to offer both an open source and a commercial license to its users. The open source license, usually the GPL, gives users the traditional freedoms to run, study, modify, and redistribute the vendor's software, while the commercial license typically provides users with the ability to embed the software in a commercial product, and may also include indemnification against intellectual property claims.

4.4 "Open core" models

As noted above, many companies have adopted a business approach in which they build proprietary software that goes above and beyond what is offered in the associated community-based FOSS version. As a rule, one can gain access to both the community edition and the commercial versions from the vendor's website.

In some cases, the differences are quite substantial. The commercial version of SugarCRM, for example, has a different user interface from the community edition, though the underlying storage mechanisms allow users to move from the community edition to the commercial one. EnterpriseDB offers Postgres Plus with numerous additional enhancements beyond the community-based PostgreSQL. Both companies offer paid support services for the community editions as well as for their own open core offerings.

4.5 Physical products with embedded FOSS

FOSS can also be used as part of a hardware/software product, with the software embedded in the hardware. Because of the licensing requirements on open source software, the product manufacturer must make the [potentially modified] FOSS code available to anyone who is interested. The manufacturer creates a business based on selling the manufactured product with no intent to generate revenue from FOSS. In this case, the manufacturer is simply taking advantage of FOSS to speed development of the embedded product.

There are numerous examples of products built on open source software. One is Amazon.com's Kindle reader device, built on Linux, with the source code for the relevant packages available on Amazon.com's website. Many Linksys routers also include FOSS software, which is available for each router, in keeping with the requirements of the GPL license, under which the code was licensed. (http://support.linksys.com/en-us/softwarenotice). The best example of embedded FOSS is Google's widely-used Android operating system, with source code available at http://source.android.com.

Many open source software components have been developed specifically for use in embedded systems, including versions of Linux optimized for real-time and embedded systems. FOSS can also be found in automotive systems and medical equipment. The Linux Foundation has recently launched the Automotive Grade Linux Work Group, and there are numerous implantable medical devices that use FOSS [16].

4.6 Documentation, training, and conferences

Thousands of books have been written on FOSS, primarily on the use of specific FOSS projects, such as Linux, but also on languages, e.g., PHP, and development environments, e.g., Eclipse, used in creating those projects.

Several publishers, most notably O'Reilly Media, Apress and Packt, have focused on FOSS projects and development tools, adding eBooks to a substantial catalog of physical books. O'Reilly's online training offerings complement their books. Adding in the O'Reilly open source conference (OSCON) makes it clear that O'Reilly's business is strongly dependent on FOSS.

There are many other conferences related to FOSS topics. Some are organized by the various Foundations to bring together people working on their projects. Others are organized to bring together people in a city or country. Beyond those are commercially organized events that make a profit from vendor exhibits and attendee registration.

4.7 Promotional materials

Last, but not least, is the market for promotional materials, including t-shirts, mugs, stickers, and other items related to various FOSS projects, conferences, commercial FOSS products, and other topics.

5 Future Directions

In the past fifteen years, FOSS development and use has grown exponentially, and now is a factor in a wide range of software development and evaluation decisions by companies and governments. Many companies have established policies for contributing to FOSS projects, as well as using FOSS in their business and projects. Licensing issues are more clearly understood, and the extent of misunderstanding around FOSS has dropped sharply, particularly as organizations become aware of how pervasive it has become. Rather than being a risk, FOSS is now often perceived as an advantage for a company, allowing them to concentrate on the unique features of their products, and building upon a FOSS infrastructure.

Beyond that, most of the largest software and systems businesses have a substantial investment in FOSS and contribute to numerous FOSS projects and foundations. Many commercial FOSS companies have built up substantial customer bases, which typically include well-known companies among their customers. All of these factors combine to make FOSS a safe business decision. These trends are likely to continue.

Beyond that, there is a growing trend of companies contributing to FOSS projects. In some cases, companies contribute people and/or code to projects as a way of supporting the project. Also, companies have contributed projects to the community with no expectation that they will receive any compensation for that code. Facebook, for example, has contributed the Cassandra data management system to the Apache Foundation, where it is now a community-supported project available to all [17]. Numerous companies have contributed code, as well as effort, to OpenStack [18], a massively-scalable operating system for private and public clouds.

There are some uncertainties ahead, though. In Section 4.2, we noted the growing use of hosted applications (SaaS) and development platforms in the cloud. Some SaaS applications, such as Google Apps and Yahoo News, were largely built on FOSS, while other hosted applications, exclusively use proprietary code. Hosted applications make the use of FOSS less visible, since the average user cannot easily tell which applications are built with FOSS.

With FOSS being less visible in these hosted applications, some vendors have instead promoted "open APIs" or platforms, a set of programming and messaging interfaces that provide limited access to the data and functionality developed by the vendor. These vendors, including salesforce.com, Facebook, and SugarCRM have all provided ways through which new projects and services can be connected to the underlying platform. While approximately half of the projects in SugarForge use an open source license for their SugarCRM add-ons, that percentage is much lower for force.com and Facebook Connect, which are built on proprietary code. In any event, open APIs are not the same as FOSS, and are under the control of a vendor who may change or eliminate them at any time.

At the same time, though, the infrastructure of cloud-based applications, such as OpenStack, is increasingly based on FOSS. Modern applications are increasingly dependent on managing large volumes of data, and need the scalability of computing and storage resources to run efficiently. New tools for large-scale data management, including Hadoop, MongoDB, and Cassandra are all FOSS. Many vendors of proprietary hosted software are making extensive use of FOSS for their hosted services.

The other important trend is the ongoing transition toward mobile devices, with their own operating systems and applications. With the notable exception of the Android platform, mobile computing has until now been dominated by proprietary software, particularly at the application level. Recently, though, mobile versions of FirefoxOS and Ubuntu have been introduced for mobile devices. It is much too early for products built on those platforms, and thus too soon to see if those products will gain a significant market share. However, the availability of a mobile Ubuntu platform will provide a highly effective pathway to bring a large body of FOSS to a mobile device, and thus an important contrast to the dominance of proprietary apps on current mobile devices.

In conclusion, community and commercial FOSS will continue to coexist with proprietary software, with a growing range of commercial activity, e.g., support and product add-ons, around community-based FOSS. It's quite likely that hosted applications will make greater use of FOSS as application frameworks and platforms become more stable and established. Proprietary vendors have changed their business practices because of FOSS and the Internet [6] and these changes are likely to continue. The end result will be a wider range of options for those who are developing future applications.

Acknowledgments

The author is grateful to Martin Michlmayr, Simon Phipps, Dirk Riehle, Giancarlo Succi, and the anonymous reviewers of an earlier draft for their valuable comments on an earlier draft of this paper.

References

[1] Cusumano, M. *The Business of Software*. Free Press, 2004.

[2] Gay, J and R.M. Stallman. Free Software, Free Society: Selected Essays of Richard M. Stallman. CreateSpace, 2009.

[3] Open Source Initiative. The Open Source Definition. http://opensource.org/docs/osd Accessed on 9 July 2013.

[4] McHugh, J. For the Love of Hacking. August 10, 1998. http://www.forbes.com/forbes/1998/0810/6203094a.html Accessed on 10 July 2013.

[5] Coverity Corporation. Coverity Scan: 2011 Open Source Integrity Report. 2012.

[6] Wasserman, A.I., How the Internet Transformed the Software Industry. *Journal of Internet Services and Applications*, vol. 2, no. 1 (2011), 11-22

[7] Lampitt, A., "Open-Core Licensing (OCL): Is this Version of the Dual License Open Source Business Model the New Standard?" http://alampitt.typepad.com/lampitt_or_leave_it/2008/08/open-core-licen.html, Accessed on 5 March 2013.

[8] Aslett, M. "Andrew Lampitt Defines Open Core Licensing". http://blogs.the451group.com/opensource/2008/09/01/andrew-lampitt-defines-open-core-licensing/ Accessed on 5 March 2013.

[9] Krishnamurthy, S., "An Analysis of Open Source Business Models". In Feller, J. et al, *Perpsectives on Free an Open Source Software*. MIT Press, 2005, pp. 279-296.

[10} Fitzgerald, B. "The Transformation of Open Source Software, " MIS Quarterly, vol. 30, no. 3 (Sept., 2006), 587-598.

[11] West, J. "Value Capture and Value Networks in Open Source Vendor Strategies," HICSS 2007. 40th Annual Hawaii International Conference on System Sciences, 2007, 176.

[12] Fink, Martin. *The Business and Economics of Linux and Open Source*. Prentice-Hall PTR, 2003.

[13] Wasserman, A. I. (2009). "Building a Business on Open Source Software". In C. Petti, *Cases in Technological Entrepreneurship: Converting Ideas into Value*. Northampton, MA: Edward Elgar. pp. 107-121

[14] Riehle, D. ,"The Single-Vendor Commercial Open Source Business Model," *Information Systems and e-Business Management* vol. 10, no. 1. Springer Verlag, 2012, 5-17. Republished from "The Commercial Open Source Business Model", *Value Creation in e-Business Management,* LNBIP 36. Nelson, M.L. et al., Eds. Springer Verlag, 2009, 18–30.

[15] Chou, T. *The End of Software*: *Transforming Your Business for the On-Demand Future*. Sams Publishing, 2004.

[16] "When Code can Kill or Cure," *The Economist, Technology Quarterly, Q2 2012*, June 2, 2012. http://www.economist.com/node/21556098. Accessed on 3 March 2013.

[17] The Apache Cassandra Project. http://cassandra.apache.org/ Accessed on 5 July 2013.

[18] OpenStack. http://openstack.org Accessed on 3 July 2013.