IDEA GROUP PUBLISHING

701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

ITP5069

Open Source Software: Opportunities and Challenges

Sofiane Sahraoui
School of Business and Management, American University of Sharjah, P.O. 26666—Sharjah—United Arab Emirates,
ssahraoui@ausharjah.edu, TEL: 00971-6-515-2346, FAX: 00971-6-515-5065

Nour Al-Nahas
School of Business and Management, American University of Sharjah, P.O. 26666—Sharjah—United Arab Emirates,
TEL: 00971-6-515-2346, FAX: 00971-6-515-5065

ABSTRACT

Free/Libre Open Source Software (FLOSS) is an imposing business reality but only an emerging discipline of business research. Critical questions have been so far only tangentially investigated and were mostly left to IT columnists and consultants to deal with, generally in a partisan fashion. Such important issues include the viability of FLOSS business models, the impact of FLOSS on the software industry structure and competitiveness, the FLOSS-based national IT strategies, the role of governments and other lobbying groups in promoting or combating FLOSS, and finally the vital matters of TCO, security, and performance among others. This paper tries to lay the foundation in analyzing the FLOSS phenomenon by pointing to the real opportunities that lie ahead and the critical challenges that have to be addressed before FLOSS can claim its rightful stake in the software industry.

OPEN SOURCE SOFTWARE: A DEFINING FRAMEWORK

FLOSS has been described as both a philosophy and a process for software development and distribution (Morgan, 2002). As a philosophy it describes a set of beliefs on how software (i.e. knowledge) ought to be developed and transferred. This philosophical discourse is generally underlined by a moral stand and is opposed to the dominant discourse of commercial software vendors who seek to make profits from the development and distribution of software. Accordingly software needs to be designed to be free itself and to enable the free sharing of all types of information (Maguire, 2003c). As a process, open source proponents advocate the superiority of a software development model that harnesses the collective intelligence of a community of users scattered all over the word (i.e. innovation communities) in opposition to the proprietary development model which relies on the efforts of a research and development team (Hippel, 2001). Abundant research dealt with the concept of "innovation communities", a term that has been coined to describe the programming efforts deployed in open-source projects (Meyer, 2003; Franke & Hippel, 2003; Von Krogh et al.; 2003; Hippel, 2001).

An appropriate definition of open source software is the following:

Open source software is software for which an executable version is made available along with the source code and which, can be modified by those who acquire it either freely or commercially.

This definition is comprehensive enough and covers most licenses, commonly termed as open-source. It also covers both software that is made available for free and software that is commercially sold. Finally it covers software that is developed by a community of users outside of organizational boundaries (i.e. innovation communities), as is typical of open source projects as well as software that is developed by software manufacturers.

IS OPEN SOURCE SOFTWARE FREE?

Richard Stallman, the initiator of the FLOSS movement, and founder of the *Open Source Foundation* developed his argument for the free flow of software and equated the open source movement to other progressive movements that sought to instill free speech and other higher moral values in society (Evers, 2000). Hence *free* has always meant freedom rather than *price zero*. More specifically, free relates to the users freedom to run, copy, distribute, study, change, and improve the software (Cervone, 2003). An extreme view goes as far as decrying any efforts to sell any programming code, or claiming any intellectual property over programming code. Software hence should not exist as an industry (Cervone, 2003).

However by far and large, it is admitted as a common practice within the open source community, and under a variety of licenses that FLOSS can be commercially sold and licensed as in proprietary commercial licensing, albeit with major differences that will be explained later. Indeed while open-source licenses are free, many software vendors have developed successful businesses selling FLOSS. *RedHat* and *SuSE* are two prominent *Linux* vendors that sell software both under the GPL and proprietary licenses. *SUN* has developed an office suite that is perceived by many as a potential threat to *Microsoft Office* suite. *MySQL*, an open source database, offers both a free-source license and a traditional commercial license (Darrow, 2003). In order to better understand what seems to be a contradiction on the surface, it is mandatory that copyright and licensing schemes for FLOSS be explained.

WHO OWNS THE INTELLECTUAL PROPERTY RIGHTS OF OPEN-SOURCE SOFTWARE?

There are many different types of open-source software licenses in use, nearly as many as fifty (see: http://www.opensource.org/licenses/). Nonetheless and as a group FLOSS licenses could be clearly distinguished from conventional proprietary licenses. The latter are generally designed to take away the user freedom to share and change the software, which is the object of the license. By contrast, open-source licenses explicitly guarantee the freedom to share and change software without any permission from its original owner (Evers, 2000).

The *GPL* (General Public License) is the most prominent license because the most visible open-source product, *Linux*, is distributed under its terms (Evers, 2000). Not only does the *GPL* guarantee the freedom to share and change software but also requires than anything linked with the concerned software be distributed as free software as well. This is known as the 'virus' effect (Evers, 2000). A consequence of this is that any software that is developed based on the *Linux* Kernel for example has to be shared back with the open-source community, hence released under a *GPL* itself. This has had a very positive consequence on the development of *Linux* as a major player in the server market and even as a serious contender to dethrone *Windows* from its desktop hegemony. Conversely, other operating systems that are distributed under open source licenses that do not require re-channeling changes, such as the *BSD* license (Berkeley Software Distribution), have led to dispersed programming efforts and even an appropriation of open-source code by proprietary vendors. *Apple*'s latest version of its proprietary operating system (i.e. *MacOS X*) is heavily based on *Darwin*, a code that is freely available under the *BSD* license.

IS OPEN SOURCE SOFTWARE BETTER THAN PROPRIETARY ONE?

In what follows, we sort the potential benefits and advantages of FLOSS over proprietary software as well as point to some shortcomings and whether they could or would be alleviated.

Ownership

When users acquire software through an open-source license, they truly become owners of the software, which means they inherit the right to modify it, share it, redistribute it, and even resell it if they have made significant additions, which they can license themselves. Software in the open-source licensing model is almost treated like a commodity rather than intellectual property that has to be traced back to its original author. Proprietary licensing restrictions do not allow any of this.

Customization and Scalability

As users are allowed to tinker with the source code, they have the possibility to tailor open source applications to their specific needs and even to redistribute the changes, including on a commercial and proprietary basis, depending under which license they have acquired the software. One offshoot of customization is scalability, which refers to the ability to run and adapt software for various IT platforms and architectures. For example, *Linux* can be installed on virtually every architecture including *x86*, *Intel Itanium*, *AMD64*, *RISC*, *PowerPC*, huge clusters, supercomputers, PDA's as well as hardware that has come out of date. On the other hand, *Windows* is restricted to very few platforms (Wheeler, 2003). *Linux* vendors like *SuSE* for instance will provide you with an architecture-specific version rather than an all-purpose operating system like *Microsoft* does.

Performance

Many reliability and performance tests yielded results supporting the fact that FLOSS is rather functional and even a strong competitor for proprietary software in terms of reliability and throughput efficiency (i.e. processing speed). In one of these tests, *Linux* was found virtually fully crash proof whereas *Windows* had serious downtime problems (Vaughan-Nichols, 1999). *Linux* was also found to have fewer errors in its code than five other operating systems and the quality of the *TCP/IP* (*Transfer Control Protocol/Internet Protocol*) implementation in the *Linux* kernel, which is vital for networking activities, was of a much higher quality compared to commercial software (Shankland, 2003). In terms of speed, FLOSS servers (i.e. servers running *Linux* or other open-source OS) proved to be faster than proprietary servers in many occasions (Rothman & Buckman, 2003). Moreover, hardware that has become out of date in the *Windows* environment because of the increasing processing power and memory requirements of the latest versions of *Windows* itself and other proprietary applications, is being rehabilitated by significantly lower system requirements from *Linux*. For instance, *Pars Wood Technology College* in the UK has deployed *Linux* on over 700 second-hand refurbished machines, which otherwise would have been sent to the dustbin (Morgan, 2002).

Security

As the saying goes, *given enough eyeballs, all bugs are shallow* (Raymond, 1998). The simplest definition of security when it comes to the computer world is "denying unwanted/unauthorized access, damage, modification, or destruction of a [computer] system to ensure confidentiality, integrity and availability of the information processed and stored" (Rajani, 2002). Since the source code is available for open-source software, it is possible for many people to view the code, spot errors, and fix them quickly. Hence FLOSS is less susceptible to hackers than proprietary software. Indeed FLOSS transparency increases security because "backdoors" used by hackers can be exposed and programmers can root out bugs from the code (The Economist Group, 2003). Security, a top concern for software users, is increasingly proving to be the Achille's wheel of proprietary software and especially for *Windows*. A number of governments around the world are wary of repeated computer-virus attacks that target *Microsoft*'s *Windows* operating system (Yamada, 2003). The latest virus attacks by the *Blaster* and *SoBig* viruses, have indeed increased concerns about *Windows* security (The Economist Group, 2003).

Security has yet another aspect which is proving detrimental to *Windows* especially with government clients, which try to avoid relying on a single OS, vendor, or center of operation. The ministry of interior in Germany justified the country's decision to adopt FLOSS in order to raise the level of IT security by avoiding monocultures (Rajani, 2002). Likewise, the government of China has been working on a local version of *Linux*, on the grounds of self-sufficiency, security and to avoid being too dependent on a single foreign supplier (The Economist Group, 2003).

Total Cost of Ownership (TCO)

As one would suspect, the TCO debate is raging between proponents and opponents of FLOSS. Several studies have been done or are under way and have reported conflicting findings. Studies tend to favor those who paid for them (Maguire, 2003a) and findings are generally disputed (Varghese, 2003). First let us define what TCO is and then we will try to identify the most contentious points.

TCO means the total amount of money that the decision of introducing new software costs, which can exceed the selling price of the software. Other cost factors include the system preparation including hardware and other necessary software, man-hours to handle the software installation, operation, and maintenance, user training, updates, cost of migration to the new software including any changes to business processes, etc. (Evers, 2000). For example for operating systems, the price of the OS itself accounts only for 20 to 30 percent of the TCO (Maguire, 2003c). In Schools, which generally benefit from targeted pricing schemes, software is only 6 percent of the schools ICT cost (Maguire, 2003c). With this in mind, the generally lower purchasing price of FLOSS ceases to be an evident advantage.

Generally what you include in the basis for calculating the TCO is context specific, hence there are no standard formulas for calculating TCO across applications or organizations. However to say that proprietary software TCO could be lower than that of FLOSS is simply "pure nonsense" (Maguire, 2003b).

CHALLENGES FOR OPEN SOURCE

Despite the competitive advantage of FLOSS over proprietary software as evidenced above, the issue is far from settled and FLOSS is not yet at a stage where it is automatically considered in corporate software procurement decisions. This is due to a variety of factors that we try to elaborate on below:

- National, regional, and corporate procurement processes are generally biased against open source (Zieger, 2003). A *RedHat* distributor in the Middle East who had sold proprietary software in the past reported that it was much harder to approach procurement and IT managers with FLOSS than it is with proprietary software (*Yahya Al Kasssab*, personal conversation, Sep 18, 2003). In many cases, "they would not even give you the chance to see them and show them your product." This is probably due to an ill-perception of FLOSS as amateur software, but also to an anti-FLOSS aggressive marketing strategy by proprietary vendors, chief amongst them *Microsoft*.
- FLOSS needs sophisticated technical know-how, hence it would not be accessible to common users. Conversely, proprietary software would tend to have an edge in functionality and user-friendliness over FLOSS solutions (Cervone, 2003). This might have been true in the past but not anymore as FLOSS has increasingly grown in user-friendliness. Set-up and service of *Linux* for example is not a major problem anymore (Brockmeire, 2003) and the latest versions of *Linux*, *StarOffice*, or *MySQL* for example do not require any more technical knowledge to install and operate than *Windows*, *Ms-Office*, or *Oracle* respectively.

- FLOSS is primarily for back-end systems running servers and e-mail systems as found in a study by the Aberdeen Group (Brockmeire, 2003). Yet again this is more a lingering perception than a true market reality. What it really means is that FLOSS has finally established itself in the back-end, especially through *Linux* at the OS level, and *Apache* at the Web-server level, and is now venturing in the front office. Recent deals involving mass migrations from *Windows* to *Linux* (The Economist Group, 2003) at the desktop and the introduction of a credible alternative to *Ms-Office* on the desktop are positioning FLOSS at all levels of organizational computing and not simply in the back-end.
- Stiff resistance from proprietary software companies has made it all the more difficult for FLOSS to break into the software mass market. *Microsoft* and its allies have sought to discredit open-source software, likening its challenge of proprietary ownership to communism and suggesting that its openness makes it insecure and vulnerable to terrorism (The Economist Group, 2003). Lobby groups for proprietary software such as the *Initiative for Software Choice*, and the *Business Software Alliance* have been continuously questioning the ethics of free software licensing. Aggressive marketing strategies by proprietary vendors are making it difficult for FLOSS to impose itself. *Bill Gates*, Founder of *Microsoft*, seized the opportunity of the *World Economic Forum* in *Davos* to "convince" world leaders to give up their government open source strategies and renew commitment to proprietary standards. Steve Ballmer, CEO of *Microsoft*, interrupted his vacation in *Switzerland* to visit the City of *Munich* government and dissuade them from adopting *Linux* on the desktop but to no avail (The Economist Group, 2003).
- Yet another serious challenge to open source is coming from *Linux* legal snarls (Maguire, 2003c). SCO, ironically a company that issued an IPO as an open-source software venture and which developed a commercial *Linux* solution (i.e. *SCO Caldera*) has filed a one billion dollar lawsuit against *IBM*, claiming the company had taken chunks of *SCO*-owned *UNIX* code-software that in the 1980s and 1990s powered most corporate IT shops and sprinkled it into *Linux*. SCO's claim meant that *IBM*'s additions are now part of almost all versions of *Linux* (Lashinsky, 2003). If *SCO* won its case, this meant that all companies running *Linux* as an OS would have to pay licensing fees to *SCO*, or would be liable to a software piracy lawsuit otherwise. *SCO*'s case against *IBM* could cast a long legal shadow over *Linux* and over the entire open-source model of licensing software (Gartenberg, 2003). This could slow down adoption of *Linux* and open source.
- A final major challenge to open source sustainability is its business model (Zieger, 2003). As software is steadily turning into a commodity, money can be made from product sales rather than service (Bank, 2003). Indeed major applications in OS and databases are in widespread use (i.e. mass-market), the industry has settled on common standards, and new features are less important than price and performance, hence reducing the importance of the intellectual content of the software and turning it more into a commodity, just like what generic drugs have done for branded ones (Karp, 2003). How can FLOSS vendors make money from software that is essentially free and which does not require much servicing then? Vendors like *RedHat* or *SuSE* who specialize in selling open source software and most notably *Linux* will probably survive and even thrive on two fronts; the first within a web of strategic alliances with hardware manufacturers which will need development and support for their open source offerings; and second within the electronic appliance and personal computing market where very small margins can yield high profits because of the large size of the market.

CONCLUSION

Open source must prove itself in key areas, including integration, interoperability, scalability, and reliability, and the portfolio of corporate applications for the platform must grow. However it is beyond the shadow of any doubt that FLOSS is the new paradigm for software development in the 21st century. By allowing innovation by anyone, it constitutes a revolutionary structure for technical innovation (Maguire, 2003c). An increasing number of countries are adopting national open source strategies, and national leadership prompting the open source way is emerging everywhere. For instance, the free software movement in Brazil has gained momentum since the *Leftist Workers Party* took office in January 2003 (Karp, 2003). A major challenge for FLOSS however is to go mainstream, and this requires massive Marketing efforts. Licensing likewise has to be brought under control in order to focus efforts and avoid the emergence of different standards that could kill this revolution. On the academic side, researchers have to get involved to put an end to rigged debates about TCO, licensing schemes and ethics, national and corporate IT strategies, etc.

REFERENCES

1. Bank, D. (Jul 9, 2003). 'Open Source' Database Poses Oracle Threat *Wall Street Journal* (Eastern Edition), p. B1

2. Brockmeire, J. (Aug 01, 2003). *Windows* vs. *Linux*: TCO Feud Rages On *NewsFactor Netwrok* pp.1-2. [Online] Available: http://www.*Linux*enterprisenews.com/perl/printer/22012/

3. Cervone, F. (Summer 2003). The open source option *Library Journal* p. 8.

4. Darrow, B. (Jul 14, 2003). *MySQL*, Pogo *Linux* team up on database appliance *Crn* (1053), p. 124

5. Evers, S. (2000). An Introduction To Open Source Software Development. [On-line]. Available: http://user.cs.tu-berlin.de/~tron/opensource

6. Franke, N. and Von Hippel E. (2003), Satisfying heterogeneous user needs via innovation toolkits: the case of Apache security software, *Research Policy*, Volume 32, Issue 7, Pages 1199-1215.

7. Gartenberg. M. (Jun 23, 2003). Microsoft can't stifle *Linux Computerworld 37 (25)*, p. 25

8. Hippel, E. (Summer 2001). Innovation by user communities: Learning from open-source software *Sloan Management Review* 42 (4), p. 82-86

9. Karp, J. (Sep 8, 2003). A Brazilian Challenge for Microsoft; The Government's Preference For Open-Source Software May Tilt the Playing Field *Wall Street Journal (Eastern Edition)*, p. A.14.

10. Lashinsky, A. (Jul 21, 2003). Penguin slayer *Fortune* 148 (2), p. 85.

11. Maguire, J. (Sep 9, 2003a). Has *Linux* Eclipsed Open Source *Enterprise Linux IT* pp.1-2. [Online] Available: http://www.*Linux*enterprisenews.com/perl/printer/22242/

12. Maguire, J. (Sep 10, 2003b). Windows Development Cheaper Than *Linux Enterprise Linux IT* pp.1-2. [Online] Avialable: http://www.*Linux*enterprisenews.com/perl/printer/22254/

13. Maguire, J. (Sep 11, 2003c). Open Source on the Brink. *Enterprise Linux IT* pp.1-2. [Online] Available:http://www.*Linux*enterprisenews.com/perl/printer/22278/

14. Meyer, P. (2003). Episodes of Collective Invention U.S. Bureau of Labor Statistics. [Online] Available: http://opensource.mit.edu/online_papers.php/

15. Morgan, E. (Mar 2002). Possibilities for open source software in libraries *Information Technology And Libraries 21 (1)*, p. 12-15.

16. Rajani, N. (2003). Free as in Education: Significance of Free/Libre and Open Source Software for Developing Countries. [On-line]. Available:http://www.maailma.kaapeli.fi/FLOSSReport1.0.html#mozTocId13212

17. Raymond, E. (1998). The Cathedral and the Bazaar. O'Reilly & Associates, Inc. Sebastopol, USA, pp.12-40.http://www.openresources.com/documents/cathedral-bazaar/main.html

18. Rothman, J. and Buckman, J. (2003). Which OS is Fastest for High-Performance Network Applications?. CMP Media. [On-line]. Available: http://www.samag.com/documents/s=1148/sam0107a/0107a.htm

19. Shankland, S. (2003). Study lauds open-source code quality. *CNET News.com*. [On-line]. Available:http://news.com.com/2100-1001-985221.html?tag=fd_top

20. The Economist Group. (Sep 11, 2003). Microsoft At The Power Point. Econimist.com. [Online] Available: http://www.economist.com/business/printerfriendly.crm/Story_ID=2054740

21. Varghese, S. (Sep 16, 2003). Gartner findings on desktop *Linux* disputed *Smh* pp.1-3 [Online] Available: http://www.smh.com.au/articles/2003/09/16/1063625013703.html

22. Vaughan-Nichols, S. (1999). Can You Trust This Penguin? What's Wrong (And Right) With *Linux? ZD Inc*. [Online]. Available: http://web.archive.org/web/20010606035231/http://www.zdnet.com/sp/stories/issue/0,4537,2387282,00.html

23. Von Krogh, G., Spaeth, S. and Lakhani, K.R. (2003). Community, joining, and specialization in open source software innovation: a case study, *Research Policy*, Volume 32, Issue 7, Pages 1217-1241.

24. Wheeler, D. (2003) Why Open Source/ Free Software? Look at the Numbers!. [Online]. Available: http://www.dwheeler.com/oss_fs_why.html

25. Yamada, M. (Sep 2, 2003). Asian Countries Seek Windows Alternative *Wall Street Journal* (Eastern Edition), p. B.10

26. Zieger, A. (Summer 2003). Open-minded *Information week* p. 25-28