Free and Open Source in Healthcare: Enough Waste

Nuno Rua

Instituto Superior Técnico/Universidade Técnica de Lisboa
Departamento de Engenharia e Gestão, Av. Rovisco Pais, 1049-001 Lisboa, Portugal

Abstract. The process of Free and Open Source Software development is explained and compared with Proprietary Software development. The requirements and difficulties of implementing computer information systems in Healthcare environments is discussed, and some Free and Open Source Software projects in the area are cited. Some advantages of F/OSS over proprietary software for healthcare informatics environments are presented and measures to accelerate F/OSS adoption are suggested.

1 Basic Concepts

Particularly in a workshop about open source software, it shouldn't be necessary to explain (again) what Free and Open Source Software (F/OSS) is, but experience shows that often, even amongst technical experts and decision makers, there are still many misconceptions and prejudices.

Free Software is as old as software itself, but was only formally defined in 1983 with the creation of the Free Software Foundation[1], as software whose user[2] is granted the following fundamental rights or liberties [1, 2, 15]:

1. The right to use the software as he pleases;
2. The right to study how the software works, and to adapt it to his needs;
3. The right to help his neighbour by giving him copies of the software
4. The right to improve the software by changing it, and distribute those changes.

The Free Software Foundation was a reaction to the evolution of the software industry towards business models based on the licensing of intellectual property, which became the software's industry generally accepted model for decades [8]. In 1998, the Open Source Initiative[3] coined the term “Open Source Software” (OSS) to place more emphasis on the availability of the code than on the user's freedom, but for most practical purposes we can join them under the same acronym: F/OSS [2, 5, 6, 7]. Recently, the success of F/OSS projects such as. Linux, OpenOffice.org, or Apache, raised some questions on the generally accepted practises on this field. Some people,

[1] http://www.fsf.org

[2] Notice the emphasis on the user, and not the owner of the software as in proprietary software's licenses.

[3] http://www.opensource.org

Rua N. (2009).
Free and Open Source in Healthcare: Enough Waste .
In *Proceedings of the 1st International Workshop on Open Source in European Health Care: The Time is Ripe*, pages 8-17
DOI: 10.5220/0001827800080017

both supporters and detractors, see F/OSS as a political and/or philosophical statement. Others, although aware of the benefits, have difficulty in finding sustainable business models [10, 12, 14, 18, 21].

The author believes that F/OSS is part of a “paradigm change” [2, 4] in the industry of software development and commercialization, and as such, still hard to grasp due to years of predefined mind frames. After the hardware revolution brought by the PC, finally the commoditization of software is about to arrive.

2 Classic Software Development Process

Even when our professions have nothing to do with the software industry, we can share a common understanding of how software is normally produced: Companies detect what they perceive are user needs, and translate them to software specifications. They assemble software engineering departments to create software based on those requirements. When ready, software (1) is packaged, priced and promoted, and sent to market (2), either directly or through chains of distributors and value-added resellers (3). Customers eventually buy the software and use it (often, users and buyers are not the same), providing the software company directly or indirectly, with money and feedback about the software:

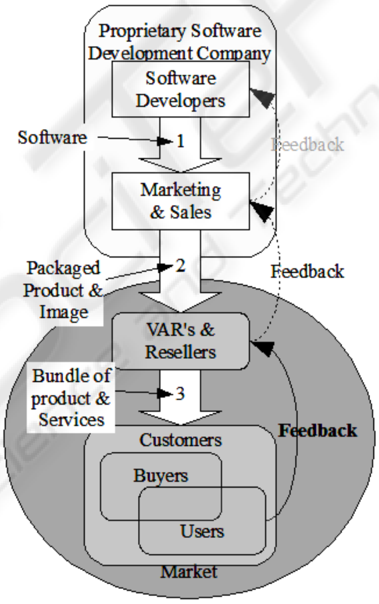

Fig. 1. Proprietary software development process.

Feedback gets thinner and distorted by all these layers, but eventually reaches the software developers, who then improve the software with bug corrections, new features, improved usability, and the cycle repeats itself. During all these phases, the source code, object of all these investments, is kept securely locked and under the company's control.

2.1 Process Critique

Figure 1 highlights the great number of layers between software development and the final users, which makes this development process very inefficient: bug correction is slow, and the new features and improvements only by chance meet what the user wants or needs [3].

Additionally, developers of proprietary software cannot use software from other sources: if it's also proprietary software, it's protected and they would incur in piracy. If it's F/OSS, most likely the result would have to be F/OSS as well. Someone who believes that can make a better job than, for example, Microsoft at its spreadsheet program Excel, cannot just pick the software and change some parts: he/she has to develop it totally from scratch, even the most basic functions. The wheel gets re-invented several times, but this seldom means more innovation, since most effort is wasted in avoiding patents or copyrighted code [6, 7, 15].

Another characteristic of software is that its cost is usually very high in the development stage, but negligible during product reproduction. This means that when a company is able to gain some market advantage over its competitors, it becomes very difficult for these to overthrow them: as more users buy the software, more money is flown into including features into that software, making it much better than competition. A larger user basis also means that is easier to exchange experiences and files, which reinforces the effect [32]. All these characteristics have turned the software market into a set of a few niche monopolies. Being well-managed companies, every proprietary software company also tries to follow Porter's recommendation of implementing barriers to entry and this translate into some sort of customer lock-in, and standards avoidance.

Even with all these limitations, this model was the industry standard for decades and, as in an Henry Ford's assembly line, all the management focus was to improve each individual step (rapid prototyping tools, lean software, market surveys, complex licensing contracts, ...) but not to change the underlying model itself.

2.2 And then, F/OSS Came Along...

The F/OSS phenomenon has been the subject of many analysis and studies, particularly in what concerns individual motivation [2, 10, 13, 18]. It's difficult to understand how so many put effort and resources into such projects. What do they gain from this?

Much research has also been done as to the applicable business models: how is it possible for a company, to make money without proprietary software licenses [12]? The fact is that, even in the classical business model, only a few companies really make money from software licenses. Most of IT companies survive by providing customers with services: consultancy, analysis and studies, bundled solutions (hardware, software and services) design, project management, punctual software development, training, support and assistance, etc. That doesn't change with F/OSS. Only the software line of the invoice gets smaller, a lot smaller.

3 The F/OSS Development Process

The general perception is that F/OSS is developed by hackers: programmers with lot of free time, that for ideological inclinations hate big corporations. They are not the type of entity that serious business should rely on. Although such cases might exist, reality is very far from this. Reality is that most successful F/OSS projects have big corporations behind them. Take Linux, with Red-Hat, Novell, IBM, or Sun. But look also at Apache, OpenOffice.org, MySQL, and many others. All these projects have large development basis, but have even larger user basis. The programmers have other motivations than just to use the software, and is not required that all users contribute with code (in fact, very few do).

Even when they don't provide lines of code, users do provide a lot of direct feedback, which rapidly improves the software: bugs are quickly detected and corrected, user needs are usually clear to developers, and normally implemented as soon as possible. The proximity of users and developers and the lack of vendor lock-in mechanisms make it very hard for companies to stay in business if they do not provide good support to their customers (remember, their core business is support and not the sale of software licenses).

Commercial companies might contribute to the code, or just provide additional services, like training and support. The communities in a F/OSS project are not a tight coherent group, but more several groups that, sometimes, intersect each other:

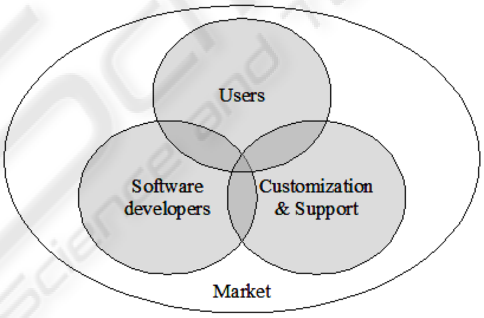

Fig. 2. F/OSS development process.

At all phases of the cycle, the code is available to everyone, inside and outside the project. To be able to be developed by several distant groups, and to take advantage of pre-existing code, F/OSS projects have to rely on strict adherence to standards, not only at the level of software interoperability, but also at the community level. This requirements is so strong that when existing standards are inadequate (either technically or due to legal constraints), F/OSS communities often create them. Many technological standards of our time are by-products of F/OSS projects: HTML, TCP/IP, SQL, W3C, ODF, ...

The development process is also very efficient, since everyone with a itch [3], can build on top of existing F/OSS code and just improve it. The rapid evolution of the code and short release cycles when compared with proprietary software might be perceived as lack of stability, but that is a false perception: there is no need to delay the introduction of a new set of features to make a big marketing campaign.

Contents are just as important as the software tools used. The emergence of Web 2.0 is a direct consequence of the success of F/OSS tools: Wikipedia, Youtube, Hi5, Google and Amazon would not be possible without F/OSS, and notice that, although relying heavily on F/OSS not all of these initiatives are free: F/OSS adoption does not mean "anti-commercial".

3.1 Process Critique

Some proprietary software companies have invested heavily in lobbying and marketing against F/OSS but their arguments can hardly be considered independent. However, some academical studies exist [19] that try to show that F/OSS is inefficient and bad for the economy. Basically their argument is that without a price mechanism to reward the best, no market equilibrium can be found.

One could reply that there are other market considerations than price, but let us just add that market theory, is a theory to explain reality. It has worked well in many situations, but should not impose itself to facts, and the facts are that F/OSS has shown that can create good, business viable, software.

Granted, the evaluation of F/OSS success is not as easy as in the classical model, where you count licenses sold and balance sheet results. For some products like Apache there are Netcraft statistics, but for the large majority, F/OSS communities are still learning how to do it [5, 14].

There are other valid critiques though: F/OSS is less focused in finishing its products than their proprietary counterparts. Marketing is poorer and user friendliness tends to have a delay of one or two years, but that sometimes also means that you can have the same functionality with one or two year's older hardware.

4 Healthcare Informatics: Not "Business as Usual"

Healthcare is a huge market of billions of euros, and healthcare informatics one of the largest pieces of this cake[4], which makes it a very appealing market to many IT companies.

The common assumption in many of these companies is that Management Information Systems and Health Information Systems are just about one and the same

[4] Sources: Gartner Databook (August 2008); IDC's and Health Industry Insight's Market Research Report (January 2007).

thing. Reality is that healthcare environments like a Hospital are far more complex than business environments, even very complex ones. Most important, this complexity is often hidden and invisible to newcomers.

This misconception results in several episodes of tremendous waste in Health informatics projects [25, 26, 27]. A major justification for these failures is the difficulty in involving users, particularly doctors, in the software development process.

Medical professionals' participation is a critical factor since healthcare problems are usually too complex to be solved by individuals, or even small teams. F/OSS projects have shown that they can flourish well in distributed settings (the Bazaar effect [3]), which makes them better prepared to capture this distributed knowledge.

Unfortunately, very few of these professionals are aware of IT technology, and even fewer are aware of the F/OSS paradigm. Information is out there, but no sales delegate or marketing campaign will bring it to them. Nevertheless, the Web 2.0 effect has already entered the healthcare setting. Patients are becoming ePatients[5]: they search the Internet for information, create forums and on-line support groups. Healthcare practitioners have to realize that they are no longer the sole repository of knowledge, and they should not oppose the wave, but ride it. How? By understanding and embracing it. This is accomplished by helping Health Professionals to learn how the F/OSS process works and how they can use it. They are not expected to learn how to code, but they need to be involved in the creation process and provide feedback.

4.1 F/OSS and Healthcare

Unlike Linux, Firefox or OpenOffice.org, F/OSS does not have large, popular, successful projects in Healthcare but, what success stories has proprietary software to show in this area? Nevertheless, some small projects were born and are growing, particularly in the areas of hospital administration, microbiology, imaging and genetics.

Some examples include:

- Vista – Openvista[6]
- Care2x[7]
- ClearHealth[8]
- i-Path[9]
- OpenEMR[10]
- CD-Medic[11]

[5] http://e-patients.net

[6] http://www.openvista.org

[7] http://www.care2x.org

[8] http://www.clearhealth.org

[9] http://ipath.ch

[10] http://openemr.net

[11] http://cdmedicpacsweb.sourceforge.net/cdmedic_en.html

- Debian Med[12]
- GNUmed[13]

These projects were able to create developers and users communities and their success will create the necessary culture for larger projects implementation [30, 31].

Sometimes, attempts to implement some type of positive discrimination in favour of F/OSS, are rejected, with the argument that public administrations should be technological neutral, but several local and national governments, and even the European Commission, are starting to recognized the potential and benefits of F/OSS in healthcare [20].

In the US, a new House of Representative Bill (HR 6898)[14] focus on electronic health information, particularly the key legal concept of health information ownership: who should own it? The government? The provider? The patient?
Other countries are also passing down legislation to foster competition on the software market, or to take benefit from the advantages of this type of software development. Spain and Sweden have similar initiatives [21, 22, 23].

Open Standards. Another important aspect of Healthcare informatics, is the utmost respect for standards (HL7 2.x, DICOM, SNOMED CT, LOINC and HIPAA) and, as we've mentioned, these are better protected in F/OSS environments. The existence of open standards however, is by no means a guarantee of success: open standards exist for decades in Healthcare, and are still not standards *de facto*. As the process of ODF and the ISO/IEC 29500:2008 approval has shown for the document format standards, the problem is not just a technical problem.

One of the issues most commonly cited regarding the difficulty of F/OSS introduction in Healthcare is the software's reliability, availability and security. These are in fact serious issues, but curiously, they are not as often cited in other, equally important healthcare fields. No one should consider the implementation of a software, F/OSS or proprietary, that does not guarantee adequate levels of reliability, stability and support. In this regard, F/OSS often offers more guarantees than commercial companies, even big ones. Companies, even large ones, are more likely to exit a line of business, than thriving F/OSS communities are likely to disappear, but even if that happens, the code is always there to be picked up and supported by someone else.

One final observation regarding the protection of private data: a fundamental aspect of healthcare practice. As the security and cryptography experts know for decades, open algorithms are always safer than proprietary ones. F/OSS doesn't just provide a viable business model: it’s the best engineering practice.

[12] http://www.debian.org/devel/debian-med
[13] http://wiki.gnumed.de/bin/view/Gnumed
[14] http://www.house.gov/stark/news/110th/legislation/200809-HIT/billtext.pdf

5 Future Trends

Fighting waste in all fronts should be a Healthcare priority at all times. Proprietary software generates monopolies, and dries innovation. F/OSS generates standards and promotes integration. The choice for Healthcare informatics industry should be an easy one.

The history of recurrent failures in Healthcare IT, particularly in large projects has created antibodies among the medical professionals. Additionally F/OSS projects typically do not have marketing campaigns and will face negative reviews from proprietary solutions vendors. The culture of F/OSS must first be taught and learned with small projects.

Due to current monopolistic market distortions, authorities should issue legislation to speed up the process of adoption of open standards, and thus promote local ecosystems of F/OSS companies. Otherwise, the future will come… from elsewhere.

6 Conclusions

The reasoning in this paper is focused on to two fundamental aspects:

1. The complexity of most Health Informatics problems are only solvable by creating communities that bring together all actors involved, and F/OSS is better equipped to achieve this;
2. Standards are not a goal per se, but a requirement to achieve interoperability in any environment, open or proprietary. When *de facto* standards happen also to be open, the costs are lower and competition in general and F/OSS in particular, is promoted.

The argument is not that all F/OSS is better than proprietary software, but that F/OSS characteristics are better suited to healthcare informatics problems.

Acknowledgements

The author's PhD investigation on Free and Open Source Software at IST - Lisbon Technical University with Prof. J. Figueiredo's supervision, is co-sponsored by Fundação para a Ciência e Tecnologia and by Rumos, S.A.

References

1. C. DiBona et al., 1999. *"Open Sources: Voices from the Open Source Revolution",* O'Reilly Media, Inc.
2. M.Cusumano et al., 2007. *"Perspectives on Free and Open Source Software",* The MIT Press.
3. E.S. Raymond, 2001. *"The Cathedral & the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary",* O'Reilly Media, Inc.

4. T.S. Kuhn, 1996. *“The Structure of Scientific Revolutions”,* University Of Chicago Press.
5. S. Weber, 2004. *“The Success of Open Source”,* Harvard University Press.
6. A.M.S. Laurent, 2004. *“Understanding Open Source and Free Software Licensing”*, O'Reilly Media, Inc.
7. J. Lerner and J. Tirole, 2004. *“ECONOMIC PERSPECTIVES ON OPEN SOURCE”* Intellectual Property and Entrepreneurship, JAI, pp. 33-69; http://www.sciencedirect.com/ science/article/B75FM-4C47MB8-3/1/7a64a45981b31c30acd7b844f3d7a6a8.
8. D. Woods and G. Guliani, 2005. *“Open Source for the Enterprise: Managing Risks, Reaping Rewards”,* O'Reilly Media, Inc.
9. K. Ven and H. Mannaert, Aug. 2008. *“Challenges and strategies in the use of Open Source Software by Independent Software Vendors”* Information and Software Technology, vol. 50, pp. 991-1002.
10. J.P. Johnson, Nov. 2006. *“Collaboration, peer review and open source software”*, Information Economics and Policy, vol. 18, pp. 477-497.
11. M. Dolores, Paula, and Salvador, Jun. 2008. *“Designing a forecasting analysis to understand the diffusion of open source software in the year 2010”*, Technological Forecasting and Social Change, vol. 75, pp. 672-686.
12. R. Rajala, J. Nissilä, M. Westerlund, Jun. 2006. *“Determinants of OSS Revenue Model Choices”,* Proceedings of ECIS 2006, Gothenburg, Sweden.
13. K.R. Lakhani and E. von Hippel, Jun. 2003. *“How open source software works: "free" user-to-user assistance”* Research Policy, vol. 32, pp. 923-943.
14. S.T. Lee, H. Kim, and S. Gupta, vol. in Press. *“Measuring open source software success”* Omega, Corrected Proof; http://www.sciencedirect.com/ science/article/B6VC4-4NX8N1G-1/1/52ca06677fde01f70459efa45491770a.
15. M. Henley and Richard, *“Open Source Software: An introduction”*, Computer Law & Security Report, vol. 24, pp. 77-85.
16. D. Cerri and A. Fuggetta, Nov. 2007. *“Open standards, open formats, and open source”* Journal of Systems and Software, vol. 80, pp. 1930-1937.
17. G. von Krogh and E. von Hippel, Jul. 2003. *“Special issue on open source software development”*, Research Policy, vol. 32, pp. 1149-1157.
18. G. von Krogh and S. Spaeth, Sep. 2007. *“The open source software phenomenon: Characteristics that promote research”*, The Journal of Strategic Information Systems, vol. 16, pp. 236-253.
19. Kalwey, Nadine; Kooths, Stefan; Langenfurth, Markus, 2003, *“Open Source-Software - Eine volkswirtschaftliche Bewertung [Open-Source Software - An Economic Assessment]”* - MICE Economic Research Studies, Vol. 4, Münster 2003
20. *“IDABC - Open Source Observatory”*; http://ec. europa.eu/idabc/en/chapter/452.
21. *“FLOSS report”*; http://www.infonomics.nl/FLOSS/ report/.
22. *“Open Source in Developing Countries - Publications - Sida”*; http://www.sida.se/sida/ jsp/sida.jsp?d=118& a=3055&language=en_US.
23. *“Software libre. Estudio del Programa IDA sobre el uso de los programas de fuentes abiertas en el Sector Público”*; http://www.csi.map.es/csi/pg5s42.htm.
24. *“Software libre: Propuesta de recomendaciones a la Administración General del Estado sobre utilización del software libre y de fuentes abiertas”*; http://www.csi.map.es/ csi/pg5s44.htm.
25. Kouroubali, Angelina. 2002. *“Structuration Theory and Conception-Reality Gaps: Addressing Cause and Effect of Implementation Outcomes in Health Care Information Systems”.* Proceedings of the 35th Hawaii International Conference on System Sciences.
26. Lorenzi, N.M. and R.T. Riley, 1995. *“Organizational aspects of health informatics. Computers in Health Care”,* ed. K. Hannah and M. Ball. New York: Springer-Verlag.
27. Sauer, C., 1998. *“Deciding the future for IS failures-not the choice you might think”.* in Rethinking MIS, W.L. Currie and R.D. Galliers, Editors., Oxford University Press: Oxford.

28. Gardner, R. Davies keynote lecture in The Computer-based Patient Record Institute Conference. 1998. Washington, DC.
29. Aarts, J. and V. Peel, *"Using a descriptive model of change when implementing large-scale clinical information systems to identify priorities for further research".* International Journal of Medical Informatics, 1999. 56: p. 43-50.
30. Sittig, D.F., *"Grand challenges in medical informatics".* Journal of the American Medical Informatics Association, 1994. 1(5): p. 412-413.
31. Forsythe, D.E. and B.G. Buchanan. *"Broadening our approach to evaluating medical information systems".* In Fifteenth annual symposium on computer applications in medical care. 1992. Washington, D.C.: McGraw-Hill.
32. Khalak, A., 2000. *"Economic model for impact of open source software".* http://opensource.mit.edu/ papers/osseconomics.pdf