ANNALS of the ORADEA UNIVERSITY.
Fascicle of Management and Technological Engineering, Volume X (XX), 2011, NR3

FREE/LIBRE/OPEN SOURCE SOFTWARE EVALUATION PROBLEM

Ivan SCEPANOVIC [1], Vesna SCEPANOVIC [2], Mira VIDAKOVIC [3]
Faculty of management, Novi Sad, Serbia,
[1] scepanovic@famns.edu.rs, [2] vscepanovic@famns.edu.rs, [3] vidakovic@famns.edu.rs

Abstract: Free Software and Open Source Software (FLOSS) development is performed by non-formal developing teams and groups. FLOSS development conditions and goals are often opposite to those in industry of property software. User choice criteria are usually determined by software quality, while numerous specialties of FLOSS development, as well as methods, and are to be considered. We believe that improved classical models and methods would give better results, especially in comparison between free and property software solutions. At this point we want to magnify the importance of FLOSS evaluation.

Keywords: Free/Libre/Open Source Software, Software Evaluation, Software Quality

1. INTRODUCTION

During the last decade, there has been a worldwide increase in the usage of Free/Libre/Open Source Software (FLOSS). There were many examples of the worldwide public and commercial applications of FLOSS. Ever increasing number of organizations are leading developers of FLOSS within various projects. Information technologies leaders like IBM, Sun, Novel, Oracle and Apple actively collaborate through FLOSS teams by exchanging knowledge and experience. There is a current trend that software market leading manufacturers are giving away to public source code of their applications in order to be increasingly involved in financial participation of development projects based on FLOSS. Apart from business organizations, many higher education institutions also use applications based on FLOSS. Software of this type is used by the teachers, administrative staff, as well as for content management systems and support for e-learning study programs. Many IT colleges and faculties worldwide based their educational programs using development tools and programming languages both based on FLOSS.

1.1. Free Software Definition

The term *free software* refers to the open source software being developed under the umbrella of General Public License (GPL). Mister Richard Stallman introduced a definition of “Free Software” and the “Copyleft concept” [19].
According to the definition, software is considered to be "free" if it grants:
- the freedom to run the program for any purpose (called "freedom 0"),
- the freedom to study and modify the program ("freedom 1"),
- the freedom to copy the program ("freedom 2"),
- the freedom to improve the program, and release the improvements to the public ("freedom 3"). [18]

1.2. Open source software (open source software)

The term Open Source Software is often used to describe Free Software. This term only relates to the disclosure of software source code, but not its freedom, so it should not be used to describe Free Software.

1.3. Representation of free software

According to the estimation of Web site SourceForge, in 2004, there has been over eighty thousand applications developed under the General Public License (GPL). [10], [9]
In February 2009, there were over 230 thousands register projects and over two million members of the Source Forge FLOSS community. What kind of a trend is it? The available data show that within a year starting from August 2008 the number of projects increased by fifty thousand and a number of community members grew by one hundred and fifty thousands [8]. There is an estimate that today there are at least one hundred and twenty thousand documented modular components in use. This gives a new impulse for the development of free software, therefore allowing programmers to further develop software that is faster and better at lower cost. Furthermore, the universal method of FLOSS evaluation does not exist yet. So, a new model should be developed for both – the end users and software manufacturers.

2. EXISTING MODELS FOR THE ESTIMATION OF FLOSS PROJECTS QUALITY

During the last ten years an increasing number of research work results were published in order to solve the problem of free software and open source software evaluation. Recently, a model for estimation of the open source software maturity has been developed. (Open Source Maturity Model, 2003.) [4]. In 2004, “Ten Rules for Open Source Software Evaluation” have been set [3] and criteria for the selection model witch measures the performance of FLOSS development projects has been established [2]. A model for assessing the software maturity called Navico Open Source Maturity Model (NOSMM) has been developed in 2005. According to this model, criteria for the maturity of software product evaluating are: quality, support, documentation, training, product integration, professional services. [6]
Open source software advisory service recommends the following ten criteria for estimating software maturity such as: reputation, ongoing effort, standards and interoperability, community support, commercial support, version, documentation (user and developer), the level of experience required to guide the use and maintenance applications, project development model, license type. [22].

3. FLOSS PROJECTS PROPERTIES THAT CAN BE CONSIDERED AS FLOSS EVALUATION CRITERIA

Besides the fact that solutions based on FLOSS provide benefits, they also carry risk factors as well as uncertainties to those who use them, because of numerous specific differences the development of FLOSS itself. Unfortunately, during the selection of software evaluation criteria it is often ignored.

3.1. FLOSS Diffusion and information technology network effect

For instance, the diffusion of Linux distribution on the server platforms has risen at the beginning of this century. The similar may be told for the FLOSS Web server application. In both cases a dynamic equilibrium has been established. On the other hand, the diffusion of free software distributions on the desktop platform is less intensive in comparison to the diffusion of server solutions. Since 2009 the research results show that only one percent belongs to the solutions being based on free software on the desktop platform.

3.2. FLOSS Impact on software industry trends

Impact of FLOSS on the software industry at the end of last century and during the first decade of this century can no longer be neglected. The Free Software Movement has initiated a new chapter in the software industry development by changing the way in which organizations evaluate, invest, develop and plan their information systems. FLOSS software brings full freedom of choice to their users. It brings freedom in terms of technology choosing as well as through independence from software vendors. In addition, the free software movement shifted the focus of software development control from developers to end users. FLOSS increases market offer and provides the basis for the replacement of monolithic, proprietary architectures with the highly modular open systems based on open source and open standards. Modular architecture provides users the ability to choose functions and adapt it to their needs. FLOSS contributes to the degree of flexibility that is unattainable using proprietary software based solutions. By removing unnecessary functions from the operating system core, user can significantly reduce the hardware requirements, and therefore reduce the cost of hardware.

3.3. Users impact on the FLOSS development

The influence of end-users to the software product development is stronger than ever. The concept of free software provides an opportunity for end users to more actively participate in the design, development and maintenance stage as well as in the software products testing. Developments in proprietary software are primarily driven by market conditions and maximum profit. Breakthroughs are happening at the moment dictating by the best for the company, not for users. On the other hand, the development of FLOSS is directed exclusively towards satisfying the needs of users without interest limits. New versions are published currently, with no test period - users are the best testers. Based on these findings we can conclude that an era of evolution of free FLOSS had started [22].

3.4. Impact of FLOSS on the software market

Diffusion and adoption of software applications based on FLOSS platform, causes its stronger impact on the global software market. Deploying competitive FLOSS alternatives causes increased pressure on manufacturers of proprietary software, breaking their prices in the market, and directly affects the issue of financial sustainability of the infrastructure of many companies that manufacture proprietary software (research, development, marketing, sales, testing, etc.). According to that secondary pressure on the commercial producers is made.

Many companies already offer free to use (but not free) Linux versions of their software. Such moves have the dual objective. They partially meet the needs of users. Second objective is an effort to inhibit the development of alternative free solutions. The main function of their attempt is to win market position for the time to come. Hybrid models, obtained by a combination of free and proprietary software with the monolithic proprietary systems and systems based on free software, in the future will be present. The main questions are - where a balance state will be established and in which direction will the process continue to unfold.

3.5. FLOSS project lifetime cycle

A number of characteristics affect the success of the project. They determine whether the software product is mature enough to survive and develop on the market of open source software. These features are particularly important in the FLOSS evaluation process.

In certain cases it is easier to make a prediction of the FLOSS development than proprietary software project. Companies that develop proprietary software may halt development of a number of different reasons. For large projects based on FLOSS, involving numerous developers and community members there is a low risk of sudden cessation of work. The openness of the community gives an opportunity to an individual to join or to leave the project. As long as there are individuals interested in the further development - project development lives. In a case of small projects, where the load is, in most cases, still on the project initiator or on a small number of people, there is a high risk of sudden stop of further development.
Size of project team and intensity of collaboration among its members should be considered as a software evaluation criteria.

3.6. Economic issues of FLOSS projects

Economic interest in initiatives related to FLOSS projects were pointed in the published work results [1], [12], [13], [20] and [21]. The authors agree that the lack of economic incentives is at the same time a great opportunity and a serious threat to FLOSS projects. We believe that this characteristic in a case of observed project should be considered when choosing FLOSS evaluation criteria.

3.7. Learning enviroment in innovative non-profit teams

FLOSS community members motivation were analysed in work of Zeitlyn [23], Hertel, Sven and Herrmann [7], O'Mahony [14]. Zeitlyn, according to the work of Raymond *Cathedral and the Bazaar* [15], concludes that the key elements of motivation work on free software projects is a programmers desire to learn as well as building personal reputation within the group. Hertel and colleagues studied the motivation of team members working on the Linux kernel. They recognize the impact of acknowledgement that developers get from the free software community because of their willingness to increase the time they spend on development. Survey finding was that there was no difference in the participants' motivation depending on whether or not receiving compensation for their work. Engineering aspect of the development and maintenance of free software, and the evolution of the Linux kernel development studied Godfrey and Qiang [5]. Survey sample consisted of 96 different versions of the core of which 34 stable and 62 development versions of the observed almost linear dependence on the number of lines of code in the core observation period (1994-2001).

For developers who want to gain experience in global software development teams, FLOSS projects can, because of its open organizational structures based on voluntary contributions and individual participants, provide excellent learning conditions and opportunities (Kang and Han [11] and Singh [17]). In fact, the conditions for learning and the ability to acquire knowledge and skills through participation in OSS projects are the main reasons why the developers decided to join FLOSS projects (Hertel et. all [7]).

Results of recent studies of learning to OSS projects give us different results: Singh [17] shows that developers learn through interaction with co-workers on the project through participation in discussions related to the forums. On the other hand, Shah [16] published the results of interviews with FLOSS developers from which he concludes that the negligible learning takes place through their participation in OSS projects. Various research results and the gap that occurs in the literature suggests that motivation and learning in the FLOSS community are not well understood in theory.

4. CONCLUSION

To access the FLOSS evaluation we have to identify, understand and appreciate all the special features of this type of software development. Literature generally gives a head to head comparison of the types of applications (web servers, content management systems, systems for data base management, etc.). Results from these, various surveys can not be compared. The changes are happening fast, so the validity of these results relate to the short time interval. There is a strong need to create a general method for the FLOSS evaluation. This model can be used by those who create software, but also by individuals and organizations when making decisions about selection of software solutions based on FLOSS.

Software maturity estimating models and the evaluation of proprietary software models exist, but cannot be applied to FLOSS. The reason for this claim lies in the fact that the teams that are developing these two different types of software are driven by different motives. Also, projects are financed in different ways and from different sources. Teams that are developing proprietary software are guided by market-oriented goals set by the strategic plans of the company that invests and/or develops software, market situation, and, on the other hand, according to user needs.

An additional problem of evaluation is the fact that Open Source software in one of several different types of software (the ownership). It is clear that these differences must be considered when creating a model of evaluation. Since existing models does not include all of these criteria we believe that they are exposed to the stated issue and deserve attention.

The quality of software developed within the FLOSS project depends on several factors. Studies have shown that one of the key criteria affecting the quality of products is motivation of participants. Also this requirement is not present in the production of proprietary software. FLOSS is generally developed in a completely open environment. The information is public. The FLOSS development is transparent in difference to proprietary software. FLOSS-based applications are different sizes - from very small packages to large complex systems. Choosing software solutions and platforms are very important issue for individuals and especially for organizations. What characteristic of FLOSS should be considered as criteria for evaluation and software products selection? What criteria are characteristic of FLOSS products? The activity of the team, the period between the two versions, supporting documentation, user support, functionality, modularity, security, monitoring standardization of document formats and possibility of integration with other products are just some of the possible criteria that can be used in the evaluation of FLOSS. The aim of this paper was not identifying the criteria, but to signify the necessity of creating an optimal FLOSS evaluation model.

REFERENCES

[1] Bonaccorsi, A, C. Rossi, Why Open Source Software Can Succeed. Research Policy. 32: 2003. pp. 1243-1258.
[2] Crowston K, Annabi H, Howison, J., and Masango, C., Towards A Portfolio of FLOSS Project Success Measures. In Collaboration, Conflict and Control, The 4th Workshop on Open Source Software Engineering, International Conference on Software Enginnering (ICSE 2004), 2004. pp. 29-33. The article is available at the following Web site: http://opensource.ucc.ie/icse2004/Workshop_on_OSS_Engineering_2004.pdf.
[3] Donham, P., Ten Rules for Evaluating Open Source Software, Point of view paper, Collaborative Consulting, 2004. The article is available at the following Web site http://www.collaborative.ws/leadership.php?subsection=27
[4] Duijnhouwer, F., Widdows, C., Capgemini Open Source Maturity Model, available at http://www.capgemini.com/technology/opensource/solutions.shtml
[5] Godfrey, W., Qiang T. 2000th Evolution in Open Source Software: A Case Study. Software Maintenance. Proceedings. International Conference, 11-14 May. 2000: pp. 131-142.
[6] Golden, B., succeeding with Open Source, Addison-Wesley/Pearson Education, 2005. Wheeler. D. How to evaluate Open Source / Free Software (OSS / FS) Programs The article is available at the following Web site: http://www.dwheeler.com/oss_fs_eval.html.
[7] Hertel, G., Sven N., Herrmann S,. Motivation of software developers in open source projects: an Internet-based survey of contributors to the Linux kernel. Research Policy. 32: 2003. pp. 1159-1177.
[8] http://apps.sourceforge.net/trac/sitedocs/wiki/WhatisSourceForge.net
[9] http://creativecommons.org/
[10] http://sourceforge.net/
[11] Kang, K. and Hahn, J. "Learning and Forgetting Curves in Software Development: Does Type of Knowledge Matter?". International Conference on Information Systems, 2009, Phoenix, AZ, U.S.
[12] Lerner J, Tirole, J., Some Simple Economics of Open Source. The Journal of Industrial Economics. L (2). 2002
[13] Nilendu, P., Madanmohan, T. R., Competing on Open Source: Strategies and Practice. available at http://opensource.mit.edu/papers/madanmohan.pdf), 2003.
[14] O'Mahony, S. 2003rd Guarding the commons: how community managed software projects protect their work. Research Policy. 32: pp. 1179-1198.
[15] Raymond, E., The Cathedral & the Bazaar, O'Reilly Media, Sebastopol, CA., 1999.
[16] Shah, S.K. 2006th Motivation, Governance and the Viability of Hybrid Forms in Open Source Software Development. Management Science (52:7), 2006, pp. 1000-1014.
[17] Singh, P.V., Youn, S, and Tan, Y. Developer Learning Dynamics in Open Source Software Projects". Forthcoming in Information Systems Research, 2010.
[18] Stallman R. M., Free Software, Free Society: Selected Essays of Richard M. Stallman, GNU Press, Boston, MA USA, 2002. pp. 83.
[19] Stallman R. M., Why Software Should Not Have Owners, is available on the Web at http://www.gnu.org/philosophy/why-free.html
[20] West, J. Dedrick. J. Open source standardization: The rise of Linux in the network era. Knowledge, Technology,f & Policy. 14 (2), 2001. pp. 88-112.
[21] West, J. How open is open enough? Melding proprietary and open source platform strategies. Research Policy. 32., 2003. pp. 1259-1285.
[22] Wheeler. D. How to evaluate Open Source / Free Software (OSS / FS) Programs The article is available at the following Web site: http://www.dwheeler.com/oss_fs_eval.html
[23] Zeitlyn, D., Gift economies in the development of open source software: anthropological reflections. Research Policy. 32: 2003. pp. 1287-1291.