The Impact of Multiple Related Projects on the Acceptance of Open Source Software

Joseph Nwankpa College of Business, Kent State University, 475 Terrace Drive, Kent, OH 44240, 330-672-1164, jnwankpa@kent.edu
Yaman Roumani College of Business, Kent State University, 475 Terrace Drive, Kent, OH 44240 330-672-1164 yroumani@kent.edu

ABSTRACT

The open source development model was seen by many as the model that would reshape the software industry as it boasted unique and unconventional characteristics that traditional development model could not match. However, despite all its potentials, the acceptance of open source software remains moderate or at least failed to keep pace with the initial expectations. The paper argues that multiple related projects made possible under the open source development model dilute the synergy that defines open source philosophy and implicitly the quality and acceptance of open source software. Rather than facilitate open source adoption, multiple related open source projects lead to lower quality of software, and negatively affect the reputation of open source software.

Keywords: Open Source Software, Quality

INTRODUCTION

Open source software is made freely available to all and is distributed under a licensing arrangement that allows the users to view it, modify it and redistribute the modification to the developer community. Open source software development projects are online based communities of software developers who voluntarily collaborate to develop software [3]. OSS development process is radically different from traditional software model. The main different is that projects were managed by large number of volunteers working freely at their chosen task [6], contributors were not geographically bound hence creating a wider developer base, and there is no explicit system-level design [10] hence encouraged flexibility and innovation. Conversely open source software seemingly became the solution to companies seeking to optimize their IT budget while re-examining the return on investment. It was heralded at a time as the gateway to adoption for companies looking to save huge licensing cost.

However, despite all these potentials, the acceptance of open source software has remained moderate or at least failed to keep pace with the initial expectations. At the user end, open source software continues to struggle to attract user-base. On the one hand, proponents of open source model are quick to blame the big corporations and their propriety wall as barriers to greater utilization of OSS. On the other hand, some argue that poor user interface, improper documentation, feature-centric development and programming for self are potential factors that hinder open source acceptance to the general public.

This paper take a different path by examining how multiple related open source projects affect the acceptance and adoption of open source software. A typically software category can have as many as a dozen variation within the open source community. Sourceforge the dominant open source project hosting

service lists more than150, 000 projects and more than 1million registered users. These projects span the open source community with little variable creating a situation where similar project with limited capabilities are developed and maintained within the open source community. Thus this paper argues that multiply related projects dilute the synergy that defines open source philosophy and implicitly the quality and acceptance of open source software.

The paper is organised as follows. In the next section we define the terms we use throughout the paper, present the relationship between multiple related project and open source adoption. We then outline the propositions and factors that influence open source adoption. Finally, we discuss the implication for future research.

OPEN SOURCE PROJECTS

Open source projects are typically initiated by individual or small groups with an idea for something that is interesting them. The common goal of an open source project is to create software that is useful or interesting to those who are working on it rather than to fill a commercial void [2]. Conversely, the open source development project may indeed be based on what developers find interesting rather than what is essential to wide-ranging users [12]. The developers of the project are the potential users of the software hence these projects are driven by a need to use basis most of the time. Usually, the project initiators generally become the owners of the project. Open source projects typically engage in no active recruiting beyond simply posting their intended goals and access address on a general public website such as sourceforge and freshmeat. As a result, many open source projects have been created because some developers felt the existing project could not satisfy their software needs.

Due to the community approach to open source projects, it is inevitably that a gap will exist between community developers who are typically technology savvy and non-developers who may not contribute codes, but are in need of software that provides a specific service [13]. If this is the case the question becomes to what degree can the software developed within the open source community accommodate the general users need? Many projects end up with software projects that satisfy the developers need, yet are so far from the need of programmers outside the project let alone the general users. The quality assurance component of open source is predicated on developers working together as a community and making positive contribution. In his paper "The Cathedral and the Bazaar" Raymond argues that high level of quality demonstrated by open source software is partly due to the high degree of peer review and user involvement. Open source software claims methodological superiority over proprietary model because of its ability to attract programmer from across the globe [5]. While most researcher on open source are quick to point on successful projects such as Apache, Linux, however, the premise that not all projects have achieved success or high quality remain a valid one [5]. Some abandoned projects and software of low quality remain visible.

TECHNOLOGY ACCEPTANCE

Technology acceptance has been widely investigated within the Information System literature. For instance, the technology acceptance model (TAM) has been widely applied to understand the attitude one has about the use of technology and subsequently, used to predict the adoption and use of information technology [1]. TAM suggests that two particular beliefs, perceived usefulness and perceived ease of use influences user's computer acceptance behaviour. Perceived usefulness is defined as the prospective user's probability that

using a particular system would enhance his or her job performance. Perceived ease of use refers to the degree to a prospective user believes that using a particular system would be free from effort [1].

Perceived usefulness of a project within the open source community can be measured by the size and duration of the project. This is usually the case as more interesting projects with higher potential will attract more developers and are usually more sustainable. Moreover, a study found that the number of developers associated with a project was positively correlated to the age of the project [4]. The study also found that projects with more developers were viewed and downloaded more often. However, perceived ease of use is more complicated and difficult to attain using the open source model. One major shortcomings of open source software is its inability to create user friendly platform. Arguably, this is because open source model was designed to serve users who are computer savvy hence did not required an elaborate user friendly design. However, as open source software to moves toward mainstream users; these concerns become evident limiting open source ability to compete favourability with proprietary software.

MULTIPLE RELATED PROJECTS AND OPEN SOURCE ADOPTION

Open source software development model has always been based on the assumption that open source community will be able to attract developers with the necessary skills, knowledge. In addition, these developers are willing to dedicate their expertises to the community at zero cost [7]. The community is able to attract such developers because of common interest existing within the open source community. Such interest can be identified by the open source project goals and descriptions. The open source community often claims that it has advantages over proprietary model of software development because of its ability to attract excellent developers across geographical boundaries. However, open source software faces certain challenges that are unique to this model. For example, the quality advantage can only be sustained if such projects indeed attract the expected developers. Due to the voluntary nature of open source model it is very unlikely that project participant will always meet this expectation. Further exacerbating this situation is the difficulty in identifying the expertise of such developers. While many open source development projects have quality practices such as bug tracking, version control system and processes designed to enhance quality yet such quality measures can only be subject to the knowledge of developers within the project. The number of projects running simultaneously as indicated in sourceforge and other hosting site is on a continuous rise while the adoption open source software has not demonstrated any significant increase to support such a rise. The continuous increase in multiple and similar projects will inevitability lead to developers spreading across these projects subsequently limiting the number of developers working on these projects. Hence we propose:

Proposition 1: Multiple alternatives of open source projects will likely decrease the number of developers and experts committed on each project.

The high quality claim attributed to open source software is due to rigorous peer review and user involvement associated with the development [9]. In addition, large followers and diversity as in the case of open source should accelerate the identification and the fixing of bugs [11] compared proprietary software development that is typically operated with limited work force. For example, open source community claim that because “many eyeballs” look for problems [9] superior software are developed through open source methodology. However, when the number of multiple related projects increases, the rigorous peer review and scrutiny claim becomes difficult to substantiate. In such situation, proprietary software is likely to come up top on quality due to more structured quality control procedures. Furthermore, as alternative projects

increases in the open source community, developers will to split across these projects hence limiting the amount of technical knowledge available for each project and the number of so call contributors on these projects. This trend is increasingly becoming common place within the open source community. Multiple projects that are not necessarily different from each other are easily created with very little chance of either increasing its contributor base or its sustainability. The quest to set up a new project is increasingly becoming the first option rather than a thoroughly assessment of the possibility of an existing project accommodating wider needs. The number of developers associated with a project is positively correlated to the age of the project [4]. As the number of alternative project increases, open source community will continue to witness growth in projects at the expense of quality and rigor. Consequently, the open source arena will be flooded with an array of ongoing projects that are either inactive or existing without enough developers to generate high quality product capable of competing against proprietary software. Hence we propose:

Proposition 2: The higher the number of multiple related projects the lower the quality of open source products.

Reputation has always been a problem with open source since inception. The open source model makes it very hard to attain the level of reputation usually earned by companies. Moreover the use of fear of security issues remains one point exploited by companies with proprietary software. However in the recent years the open source has made significant progress with successful projects such as Apache, Linux and Perl languages. The argument about security concerns is that since everyone can see the code, security problems are located and easily fixed. The problem with that argument is that there are countless open source projects but very few developers or dedicated security researcher with neither the time nor the patient to go over code and test for flaws [8]. With the continuous raise in open source project and declining level of peer review it is therefore inevitable that security concern will continue to linger and affect open source reputation. Furthermore, as the numbers of multiple related projects continue to even increase more, the declining number of developer will decline even further exacerbating the reputation of open source project and inhibiting its acceptance and deployment. Therefore we propose:

Proposition 3: The higher the number of multiple related projects the lower the perceived reputation open source software.

The adoption of open source software clearly is not keeping pace with the number of open source projects. Several reasons account for this moderate adoption rate ranging from lack of formal support to functional gaps in the product. While adoption has been increasing at a very moderate rate, however the number of open source projects has increased significantly. This inconsistency can be understood as many open source projects are neither interesting nor sustainable. Instances where several related projects are simultaneously developed remains very common in today's open source communities. Typically, many of these projects do not reflect the need of wide-ranging users. Adoption rate in the open source software can only be accomplished if users see the usefulness of the software and if the software has high usability. To achieve these objectives open source projects need to attract the right developers and engage in adequate peer to reviews. It is very difficult to achieve high adoption rate with multiple related projects as resources which could be utilized in a single project are divided across these related projects. Furthermore, the quality and sustainability of the project becomes a concern thus reinforcing the already existing stereotype that open source software is not reliable. Hence we propose:

Proposition 4: The higher the amount of multiple projects the lower the adoption rate of open source software.

DISSCUSION

In this paper, we argue that multiple related projects in the open source community are having significant impact in the adoption of open source software. For open source project to maintain its high quality, adequate number of developers need to be actively engaged in such projects. The lower the number of related projects, the more likely that existing projects will attract more developers leading to proper and rigours peer review. Open source software can only compete with proprietary software if it of high quality and perceived to be reliable. Quality remains of an undeniable sales point for open source project. Hence, there is need for the community to strive to sustain quality. While open source community can boast pool of talented developers however, such advantage can only be achieved if these developers work together as a group. The peer reviews and level of scrutiny typical of open source need to be clearly followed while the governance structure of project should be design to discourage the creation of multiple related projects.

We argue that open source software should be developed with wide-ranging users in mind. Open source software since it is written by programmers is based on their impression of what the world need which is not often a good indicator of average user's need. The idea that projects are developed to serve only the project contributors cannot enhance the acceptance and adoption of open source software. Without outside feedback and exposure to external user base, open source software designs will too often lag behind proprietary commercial software. Having fewer parallel projects will help the open source community to identify and implement software capable of serving the needs of wider user base. If the open source community cannot agree on a single project serving the community, it is difficult to argue that such community will have the ability to produce software capable of satisfying wide-ranging users. Moreover, it is easier for end users outside the open source community to select from fewer choices than having to select from an array of software projects. Typically, the users will tend to believe that such products lack the cutting edge and quality when compared to proprietary products.

CONCLUSION

This paper began with examining the open source model to software development and how growth in open source projects has not matched same level of growth in user adoption and use. While the open source community has witnessed tremendous increase as indicated in the number of registered users and ongoing projects, such increase does not come without some concerns. We linked the relationship between multiple related projects and open source adoption. Rather than facilitate open source adoption, we can see that multiple related open source projects lead to lower quality of software, and negatively affect the reputation of open source software. At a fundamental level these arguments invite open source community to review existing projects and licensing of new open source software. There is an overwhelming need to limit the number of related projects and to concentrate on building software capable of satisfying the need of wide-ranging user base.

REFERENCES

[1] Davis, F (1989) “Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology”. *MIS Quarterly,*1989, 13(3) 319-339

[2] Godfrey M.W and Tu Q (2000) “Evolution in Open Source Software: A Case Study”. *Proceedings of the International Conference on Software Maintenance (ICSM 2000) San Jose California* 2000, 131-142

[3] Hippel E.V and Krogh G.V (2003) “Open Source Software and the “Private –Collective” Innovation Model”: Issue for Organization Science *Organization Science* 2003, 14 (2)

[4] Krishnamurthy, S. (2002) “Cave or Community? An Empirical Examination of 100 Mature Open Source Projects” *First Monday* 2002, 7(6)

[5] Michlmayr M, Hunt F and Probert D (2005) Quality Practices and Problems in Free Software Projects. *Proceedings of the First International Conference on Open Source Systems Genova,* 11th - 15th July 2005, 24-28

[6] Mockus A, Fielding R.T and Herbsleb J.D (2002) Two Case Studies of Open Source Software Development: Apache and Mozilla *ACM Transactions on Software Engineering and Methodology* 2002, 11 (3) 309-346

[7] O’Reilly, T (1999) "Lessons from Open-Source Software Development," *Communications of the ACM* 1999, 42(4) 33-37.

[8] Ragan S (2009) Open Source getting bad Reputation on Security says Vendor. The Tech Herald www.thetechherald.com

[9] Raymond, E.S.(1999) “The Cathedral & the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary”, O’Reilly,Sebastopol, CA, 1999.

[10] Vixie, P., (1999) “Software Engineering, Open Sources: Voices from the Open Source Revolution”, first ed. O_Reilly, 1999, 91–100.

[11] Weber, S. (2004) “The Success of Open Source” Harvard University Press, Cambridge, Massachusetts.

[12] Wu, M.W., Lin, Y. D (2001) "Open Source Development: An Overview," *IEEE Computer,* 2001, 33-38.

[13] Zhao, L and Elbaum, S (2003) “Quality Assurance Under The Open Source Development Model” *The Journal of Systems and software* 2003, 66 65-75