Reducing Knowledge Loss in Open Source Software

Mehvish Rashid
Dublin City University,
Dublin, Ireland.
Lero, the Irish Software
Research Centre, Ireland.
Mehvish.Rashid2@mail.dcu.ie

Paul M. Clarke
Dublin City University,
Dublin, Ireland.
Lero, the Irish Software
Research Centre, Ireland.
Paul.M.Clarke@dcu.ie

Rory V. O'Connor
Dublin City University,
Dublin, Ireland.
Lero, the Irish Software
Research Centre, Ireland.
Rory.OConnor@dcu.ie

ABSTRACT

Contributor turnover leads to knowledge loss in OSS projects. The structure of the OSS community is transient in nature, yet continual maintenance of OSS projects is required for their sustainability. Even though knowledge creation and sharing is abundant, knowledge is not evenly distributed among contributors. Only a small subset of contributors called core members make major code contributions in OSS projects. It is costly for a contributor to maintain code from other contributors on the project and to seek out assistance and information required, resulting in productivity loss. Knowledge retention mechanisms, we suggest, could be improved in OSS projects. The objective of our work is to integrate the concept of knowledge retention in OSS projects. The challenge is how to apply concepts of knowledge management in such a dynamic community with a transient workforce.

Author Keywords

Open Source Software; Software Maintenance; Knowledge Creation; Knowledge Sharing; Knowledge Management; Knowledge Retention.

ACM Classification Keywords

D.2 [Software Engineering]: Distribution, Maintenance, and Enhancement, and K.6 [Software Management]: Software development, and Software Process.

INTRODUCTION

Our topic of research is knowledge loss in Open Source Software (OSS). A survey in 2015 reported that nearly 66% of companies incorporate OSS with the commercial software [18]. The qualities of OSS such as freedom to use, change, and redistribute [1] with the applicable restrictions based on the license agreement [19], has made it a critical element of software industry.

OpenSym 2017, August 23-25, 2017, Galway, Ireland.

DOI string to be included here.

Knowledge loss in OSS projects is not an extensively explored area but is vital for the sustainability of OSS projects. After completing the literature review, at this stage am well prepared to test out my ideas.

Evolution of a FLOSS project, result in dynamic transient teams of contributors who are constantly joining, leaving, or changing their role in the project. The phenomenon of resources joining and leaving is referred to as turnover [6]. Turnover leads to knowledge loss in OSS projects [17]. In many large OSS projects, a high turnover have been observed leading to the formation of the succeeding development teams [17]. In order to continue with the software development tasks succeeding development teams require knowledge about the developed source code. The author of the source code has a strong relationship with the authored code. When the author of the code leaves the project and her code is abandoned, software development can halt due to knowledge loss [16]. Knowledge loss is a threat to the sustainability of OSS projects.

The objective of this research is to introduce proactive knowledge exchange mechanisms in the OSS projects for knowledge retention. Currently the knowledge acquired in OSS project communities is reactive in nature and through asynchronous communication such as mailing lists, blogs, discussion forums, and Internet Relay Chat (IRC). Our focus is on how to facilitate a more uniform distribution of knowledge in the OSS projects.

The following section provides details on the literature relevant to knowledge loss phenomenon. We conclude literature section with two broad questions followed by the section relating to the research methodology, which is still at an infancy stage. The contribution sections lists the expected outcomes and benefits of this work to the scientific community.

RELATED LITERATURE

Knowledge in this work is driven from information [3] and accumulates with individual's experience, evolving through communication and inference [21]. Knowledge is classified into two types, explicit and tacit: explicit knowledge is formalised and documented while tacit knowledge is based on interactions [8]. Contributors in OSS can take time to become productive [22]. The time required to learn the

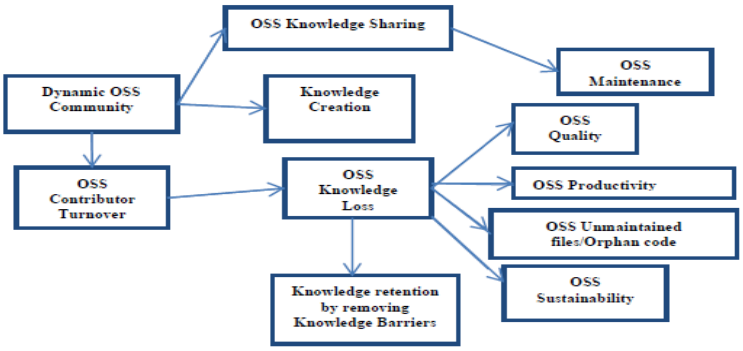

Figure 1. Mind map of related literature on OSS knowledge loss [15]

inner workings of the project, when experienced contributor leaves, causes productivity loss [9]. The structure of the OSS community is transient in nature with the participating contributors as volunteers and sometimes as paid contributors. The volunteers participate in OSS projects to learn new skills, contribute code and build a reputation within the OSS community that may benefit them in the future career opportunities [2]. The development in OSS continues in independent, self-assigned, and in parallel streams without much coordination due to geographical dispersion [11]. Mostly volunteers participate in community based projects [10].

A small subset of contributors called core members make major code contributions (80%) [12]. When knowledge distribution is among a small group of contributors, one contributor leaving can cause considerable loss of system files in OSS projects [5]. The code that is abandoned is argued to increase the numbers of reported defects as well [14]. The maintenance of abandoned code is difficult because of knowledge lacking on the code creation and structure [5, 7].

The resolution to knowledge loss in OSS projects is by Knowledge Management (KM). Two KM activities such as knowledge creation and knowledge sharing are evident in OSS projects. Knowledge creation can be related to the process of knowledge creation described by the four modes of knowledge conversion: *Socialisation, Externalisation, Internalisation,* and *Combination* are coined as *SECI* [13]. *Socialization,* which is from tacit to tacit knowledge, *Externalization* which is from tacit to explicit knowledge, *Combination* of explicit to explicit knowledge and *Internalization* which is from explicit to tacit knowledge. In SECI, knowledge is created from socialization among project members. This knowledge is made explicit or documented through externalization. This explicit knowledge is integrated with the existing explicit knowledge through combination. The explicit knowledge when acquired by an individual is again converted to tacit knowledge by internalization.

Knowledge sharing is through technology-mediated channels where knowledge is stored in code repositories, projects websites, blogs, bug reporting, and bug tracking databases, and mailing lists. Gamification [20], social media sites such as GitHub[1] for coding, StackExchange[2] and StackOverflow[3] network, play an important role in OSS projects [15]. Even though knowledge sharing on the OSS project is abundant, there is no mechanism to articulate the undocumented knowledge confined to a person. The mechanism of enabling and embedding the knowledge in an organisation is achieved through Knowledge Retention (KR). KR is essential for the sustainable performance of an organisation [4].

The above details on the problem of knowledge loss in OSS projects are captured in a mind map in Figure 1 [15].The above discussion leads to the formulation of two main research questions, expected to evolve in the future [15].

The first question will identify the effective KM practices that can be applied to OSS projects. The challenge will be adapting the practices to suit the needs of dynamic structure in OSS communities.

RQ1. Which knowledge management practices enable an effective knowledge management strategy for OSS projects?

Once the practices are assessed for their suitability to OSS communities, the next step will be their integration within the working structure OSS projects.

RQ2. How to integrate knowledge management practices with established work practices in OSS projects?

In FLOSS projects, contributors are under no obligation to notify the project community when they leave. The mechanisms of knowledge retention in an organisation are more reactive in nature, triggered when an employee is

[1] **https://github.com**

[2] **http://stats.stackexchange.com/**

[3] **http://stackoverflow.com**

leaving. Conversely, in FLOSS, it is not certain when a contributor will leave; therefore, a proactive approach to retain knowledge is required. Further, the main consideration is to introduce mechanisms that are non-intrusive and do not cause an overhead that reflects on the project's productivity.

In order for the KR process to be operational in FLOSS, Tacit knowledge, also known as informal knowledge, has to be converted to explicit knowledge. In the OSS communities, tacit knowledge is acquired through socialization on technology-mediated channels. The tacit knowledge must be converted to explicit knowledge and integrated into the existing explicit knowledge repository to minimise knowledge loss in FLOSS. The creation of knowledge is an ongoing process and therefore the retention mechanism has to be effective.

RESEARCH METHODOLOGY

OSS projects have an open access to source code, mailing lists, blogs, discussion forums and bug reporting tools. In this research the real-time data will be used along with intensive engagement with OSS projects communities. The focus will be to improve the mechanism of KR and to formulate techniques to examine and report on KR health on OSS projects. The validation will be carried out by involving OSS communities. The research methodology is yet to be elaborated but various alternatives are under consideration at the present time. The milestone of designing a research methodology is expected to be complete by the end of October 2017.

CONTRIBUTIONS

The main contributions to the scientific community are:

- The integration of KM processes in OSS projects leading to the formulation of KR practices in OSS projects.
- This research focuses on techniques that will promote a more uniform distribution of knowledge in OSS communities, further encouraging a healthy environment for knowledge exchange.
- If successful in developing a valid technique for evaluating KR on OSS projects, commercial organisations might employ such vehicles in order to assess the health of OSS projects as a mean to improving OSS selection decisions.

ACKNOWLEDGEMENTS

This work was supported, in part, by Science Foundation Ireland grant 13/RC/2094 to Lero, the Irish Software Research Centre (www.lero.ie).

REFERENCES

1. Kevin Crowston, James Howison and Hala Annabi. 2006. Information systems success in free and open source software development: theory and measures. *Software Process: Improvement and Practice*, 11 (2). 123-148. 10.1002/spip.259
2. Kevin Crowston. 2011. Lessons from volunteering and free/libre open source software development for the future of work. in *Researching the Future in Information Systems*, Springer, 215-229.
3. Thomas H Davenport and Laurence Prusak. 1998. *Working knowledge: How organizations manage what they know*. Harvard Business Press.
4. Quang Minh Doan, Camille Rosenthal-Sabroux and Michel Grundstein. 2011. A Reference Model for Knowledge Retention within Small and Medium-sized Enterprises. In *KMIS*, 306-311.
5. Samuel M. Donadelli. 2015. The impact of knowledge loss on software projects: turnover, customer found defects, and dormant files *Software Engineering*, Concordia University, 85.
6. Matthieu Foucault, Marc Palyart, Xavier Blanc, Gail C. Murphy and Jean-Rémy Falleri. 2015. Impact of developer turnover on quality in open-source software *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, ACM, Bergamo, Italy, 829-841.
7. James D. Herbsleb and Audris Mockus. 2003. An empirical study of speed and communication in globally distributed software development. *IEEE Transactions on software engineering*, 29 (6). 481-494.
8. Chris S. Hutchison. 2001. Personal knowledge, team knowledge, real knowledge. In *EUROCON'2001*, 247-250 vol.241. 10.1109/EURCON.2001.937805
9. Daniel Izquierdo-Cortazar, Gregorio Robles, Francisco Ortega and Jesus M Gonzalez-Barahona. 2009. Using software archaeology to measure knowledge loss in software projects due to developer turnover. In *System Sciences, 2009. HICSS'09. 42nd Hawaii International Conference on*, IEEE, 1-10.
10. Gwendolyn K. Lee and Robert E. Cole. 2003. From a Firm-Based to a Community-Based Model of Knowledge Creation: The Case of the Linux Kernel Development. *Organization Science*, 14 (6). 633-649. 10.1287/orsc.14.6.633.24866
11. Martin Michlmayr. 2007. Quality Improvement in Volunteer Free and Open Source Software Projects: Exploring the Impact of Release Management University of Cambridge.
12. Audris Mockus, Roy T. Fielding and James Herbsleb. 2000. A case study of open source software development: the Apache server *Proceedings of the 22nd international conference on Software engineering*, ACM, Limerick, Ireland, 263-272.

13. Ikujiro Nonaka, Ryoko Toyama and Noboru Konno. 2000. SECI, Ba and Leadership: a Unified Model of Dynamic Knowledge Creation. *Long Range Planning*, 33 (1). 5-34. http://dx.doi.org/10.1016/S0024-6301(99)00115-6

14. Tobias Otte, Robert Moreton and Heinz D Knoell. 2008. Applied quality assurance methods under the open source development model. In *Computer Software and Applications, 2008. COMPSAC'08. 32nd Annual IEEE International*, 1247-1252.

15. Rashid Mehvish, Paul M.Clarke, Rory V. O'Connor. 2017. Exploring Knowledge Loss in Open Source Software (OSS) Projects. *To appear in: Proceedings of 17th International SPICE Conference (SPICE 2017)*, Palma de Mallorca, Spain.

16. Peter C Rigby, Yue Cai Zhu, Samuel M Donadelli and Audris Mockus. 2016. Quantifying and Mitigating Turnover-Induced Knowledge Loss: Case Studies of Chrome and a project at Avaya. In *Proceedings of the 2016 International Conference on Software Engineering*, Autrin, Texas.

17. G. Robles and J. M. Gonzalez-Barahona. 2006. Contributor turnover in libre software projects *IFIP International Federation for Information Processing*, 273-286.

18. Black Duck Software. 2015. Seventy-Eight Percent of Companies Run on Open Source, Yet Many Lack Formal Policies to Manage Legal, Operational, and Security Risk. N.p., 2017. Web. 8 June 2017.

19. Chandrasekar Subramaniam, Ravi Sen and Matthew L Nelson. 2009. Determinants of open source software project success: A longitudinal study. *Decision Support Systems*, 46 (2). 576-585.

20. Bogdan Vasilescu, Alexander Serebrenik, Prem Devanbu and Vladimir Filkov. 2014. How social Q&A sites are changing knowledge sharing in open source software communities *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*, ACM, Baltimore, Maryland, USA, 342-354.

21. Michael H. Zack. 1999. Managing Codified Knowledge. *Sloan Management Review*, 40 (4). 45-58.

22. Minghui Zhou and Audris Mockus. 2010. Developer fluency: achieving true mastery in software projects *Proceedings of the eighteenth ACM SIGSOFT international symposium on Foundations of software engineering*, ACM, Santa Fe, New Mexico, USA, 137-146.