A Survey of Learning Methods in Open Source Software

Aidarbek Suleimenov, Assiya Khuzyakhmetova and Antonio Cerone

Department of Computer Science, Nazarbayev University, Kazakhstan
{aidarbek.suleimenov,assiya.khuzyakhmetova,antonio.cerone}@nu.edu.kz

Abstract. Open source software (OSS) is usually developed by heterogeneous groups of people, each with their own interests, motivations and abilities. Therefore, it is important to establish the best software development and contributing practices early in the life-time of the project. Such practices should foster the contributors' involvement in the OSS project as quickly as possible. The sustainability of an OSS project is heavily based on the underlying community of contributors and on the knowledge and skills they bring to the project and they acquire and develop through their participation in the project and interaction with the project community. Therefore, identifying and investigating contributors' learning processes is an important research area in OSS.
This survey paper presents an overview of open source learning methods in order to explore how community interaction impacts the development and application of OSS learning processes in other areas, especially in education. It is argued that collaboration with peers and consistent code contributions result in learning progress in OSS. Typical research in this area is based on case by case analysis, whereas this survey tries to highlight and combine the outcomes of several research contributions from the literature.

Keywords: Open Source Software · Learning Processes · Learning Methods · Education.

1 Introduction

Free/libre/open source software (FLOSS), or simply open source software (OSS), is software that is released together with the source code under a license that protects the right to study, change and distribute such source code. OSS is usually developed in a collaborative manner within a distributed OSS community. Members of such a community are called contributors and their contributions consist of pieces of code or documentation to the project, support provided to the software users, etc [21]. OSS is widely used today in many households, small businesses and enterprises, agricultural conglomerates, etc. It is estimated that OSS saved dozens of billions of dollars for companies [26]. Furthermore, the field is still expanding.

Important challenges in OSS development include how one can learn skills in the OSS environment. Skills can refer to technical and social skills which are required for contributing to OSS. Technical skills are those that help to make contributions in terms of code while social skills are those that help to interact with the community [5]. The open source environment requires individuals to gain skills through the process of using resources such as mailing lists, bug trackers, IRC chats, version control systems, etc.

It is important to understand learning processes in OSS projects as this helps maintain further development of the project and its support. As people learn more about OSS projects, there is empirical evidence that they become more active and efficient contributors to the community [4]. Ye and Kishida suggest that the main motivation for people to participate in OSS software development is primarily to acquire skills and learn [34], an extrinsic motivation that is normally associated with and complemented by a number of intrinsic motivations, including enjoyment, sense of creativity and accomplishment, and intellectual stimulation [3, 21]. Therefore, it is also important to investigate how one can gain skills in OSS projects in order to fulfill individual motivations, which is a significant factor of OSS development.

In this paper, we present an overview of learning processes in OSS communities and discuss the literature with a particular emphasis on formal and informal methods of learning. Section 2 clarifies the distinction between informal and formal learning both in general terms and in the specific OSS context. Section 3 reviews the literature on the analysis of OSS learning methods with reference to community interaction, code contribution, internet technologies and communication tools. Section 4 discusses how OSS skills acquisition frameworks could possibly be applied to learning in external activities in software development and higher education. Finally, Section 5 summarizes the contributions and limitations of the paper and envisages the future work in terms of both practical teaching and research.

2 Formal and Informal Methods of Learning

Formal learning refers to a structured form of learning that leads to certification by an official education organization [20]. Such a form of learning is normally highly teacher-centered as it requires some authority to conduct class sessions, provide feedback, give grades, etc. Examples of formal learning include higher education and professional certifications.

Informal learning [15, 17, 23], instead, occurs outside education organizations and has neither any set objective in terms of learning outcomes nor any specific structure. In general, informal learning is never intentional from the learner's standpoint and does not necessarily lead to certification. However, it should not be confused with non-formal learning, which often refers to organized learning outside the formal education system, such as in a short-term format and/or on a voluntary basis. There are several perspectives in defining informal learning that differ in terms of intentionality and awareness at the time of the learning experience [20]. In the context of OSS environments informal learning specifically refers to learning-by-doing or project-based collaboration. This also includes community interactions, code contributions, source code reading, technology usages, etc.

In order to make contributions to OSS, it is essential to have required technical and social skills [1]. In our survey we focus on contributions that identify how these skills can be acquired. Due to the distributed nature of OSS development and its project based approach, it could be hypothesized that OSS skills are acquired through informal learning [5]. Therefore, it is necessary to identify the specific informal learning methods that help individuals to acquire such skills.

One of the biggest challenges of this task is the fact that it is not trivial to assess the learning dynamics within an OSS environment. There are no assignments or exams to check whether contributors gained any skills or learned some concepts. Thus it is challenging to gather empirical evidences. One of the main research approaches in this area is to perform data mining analysis on OSS artifacts, such as mailing lists or code history changes [13, 22, 28]. In addition, surveys are also used [9]. In the next sections some informal methods of learning will be considered and their effects on learning will be discussed.

3 Learning Methods in OSS

Learning methods in OSS not only refer to an informal learning context but are also heavily based on practice rather on acquiring notions. Moreover, practice not only occurs at individual level but it is fostered and mediated by the participation in the OSS community and the communication and development infrastructure built around the OSS project.

3.1 Open source community interaction

Community growth around some projects could actually decrease the complexity of the system over time when compared to projects with only one individual maintainer [11], thus making community an essential part of any big OSS project. Moreover, open source communities play a vital role in contributors' skill acquisition. A specific OSS community grows around a particular project. This makes each community different from each other, thus making it hard to generalize and create comprehensive guidelines for newcomers.

The distributed nature of software development in these communities and the constant increase in software complexity make it inevitable for contributors to communicate with other peers within the community during the process of skill acquisition [5]. However, the question arises on whether or not the interaction with the community is actually beneficial for acquiring skills.

Singh et al. [28], collected data from 251 developers contributing to 25 OSS projects hosted on SourceForge over a period of six years. In their study a developer is a person that contributed at least 10 times in the period including both contributions to emails and to CVS. Then a Hidden Markov Model was built out of such data and it was discovered that learning from peers is one of the most important sources of learning in OSS. According to Wen [32] such a form of learning can also facilitate knowledge-sharing for some domain-specific skills, for example in the area of security.

Moreover, Kuk [14] found out that active interaction with the community not only increases the individual's skills but, combined with code and content contributions, also moves contributors "to the center of OSS development". Furthermore, personal experience within the community and interactions with the community have a long term dynamic impact rather than a short term static impact on a developer's code contribution behavior [28].

The "center of OSS development" refers to the core members of the OSS community, who do make the majority of contributions [29] and decisions [21]. In the study by Kuk [14], 1500 messages from two OSS development mailing lists were analyzed and it was found out that contributors with high degree of interaction with others are more likely to become core members of OSS projects. Kuk stresses that such interactions also accelerate releases of individual knowledge resources and exchange of information within the community [14].

Finally, Sowe and Stamelos [29] divide the learning process of individual actors in four phases through which knowledge evolves: socialization, in which knowledge is implicitly shared, externalization, in which tacit knowledge is made explicit to the community, combination, in which community explicit knowledge is combined and organized as abstract knowledge, and internalization, in which abstract knowledge is absorbed and further combined with individual knowledge and experiences to produce new tacit knowledge. Building on this conceptual learning model, Cerone and Sowe [7] describe OSS projects as learning and development environments in which heterogeneous communities get together to exchange knowledge through discussion and put it into practice through actual contributions to software development, revision and testing. This leads to the view of OSS communities as open participatory ecosystems in which actors create not only source code but a large variety of resources that include the implicit and explicit definitions of learning processes and the establishment and maintenance of communication and support systems [6].

3.2 Code contributions

Code contributions play a vital part in OSS learning due to the project-based nature of OSS development. Code contributions can also show the level of expertise of the contributor. On the one hand, level of complexity of the code, frequency of code contributions, domain the code was written for and number of bugs that appeared after the new code was introduced could possibly show the learning progress of the individual contributor. On the other hand, the level of expertise increases the reputation of the contributor, thus serving as a major source of motivation for developers to participate in a community [27].

There are empirical evidences that confirm the importance of code contributions and bug fixes in the learning process. For example, Kim and Jiang [13] analyzed the history of code changes of five OSS projects with overall 100 contributors. According to their study, the number of bugs resolved increases the chances for individuals to learn and reduces the chances to produce bugs in the future. This is particularly important, since it was also found out that experienced developers are as prone to introduce bugs as inexperienced developers [13].

In addition, Krogh *et al.* [30] found that it is expected for newcomers to specialize only in one area of expertise. Their study was based on the analysis of Freenet OSS project contributors' behavior. After interviewing contributors, it appeared that they made contributions on the basis of prior knowledge. There are several reasons for this. One reason is the fact that contributing code to OSS projects can be hard [12] and therefore requires extensive experience. It might be more beneficial and easier to focus on one area in order to make significant contributions. A second reason is that expertise in some particular area could be highly beneficial for the community as a whole, once such expertise undergo the process of knowledge sharing. Thus, these benefits can make the community more than willing to accept newcomers in exchange for their experience [32].

Learning through code contributions and learning from community interactions are not necessarily mutually exclusive. Interaction between newcomers and experienced project members is essential for newcomers to make code contributions. Newcomers need to be allowed to make contributions that are equivalent to their abilities and experienced members could potentially help them in identifying such kinds of contributions via collaborative efforts [8].

Finally, we can say that code production not only fosters learning through practice in the code contributor, but also drives learning in those community members who study and test that code. A similar role is played by other artifacts of the OSS production process, such as documentations, guidelines and even licenses.

3.3 Internet technologies and communication tools

OSS is usually developed by a heterogeneous group of people distributed all over the world [21], each having own role in the development of the final product. Every software engineering project depends on collaboration, and collaboration is essential in OSS. Since synchronized actions are significant for the completion of complex tasks, effective communication and collaboration play an important role in the production process [33]. Thus, developers are required to use some sort of internet technologies in order to interact with each other. In OSS communities this role is usually performed by mailing lists, internet relay chats (IRC), remote version control systems, discussion forums, bug trackers, project management tools, documentation web pages and many other tools depending on the projects' needs.

All these tools serve as a way for knowledge manifestation, which makes it unnecessary for face to face contact or for newcomers to use the bandwidth of their experienced peers in order to answer already answered questions. All these conclusions were made from an observation of K Desktop Environment (KDE) OSS project community [10]. Other kind of tools that can facilitate learning in OSS environments are Question & Answer internet communities such as Stack Exchange. Users can post questions on a wide variety of topics and answer questions of other people. Since participants are not paid for their efforts in giving answers, anyone can freely benefit from interacting with this community in a similar way to participating in OSS project communities [25].

We can conclude that the use of internet technologies and communication tools may lead to more efficient development and faster learning processes.

4 Application of OSS learning methods

There are several ways of transferring informal OSS learning methods to formal education: opening course materials and making them free to access, making students generate content for the course and contribute to its processes and usage of technologies. Weller and Meizsner [31] report on some of these ways, and analyze their effectiveness and benefits when applied to formal education.

However, the main strength of OSS learning methods is the process of collaboration and how it results in learning-by-doing. In fact, after recognizing the importance of OSS learning-by-doing in building big projects in a collaborative manner, we now consider how to apply such methods to other areas, especially to education. One of the main aspects of education in an OSS environment is the fact that both expert (teacher) and learner (student) participate in content creation and knowledge sharing [19]. Another fundamental aspect is that traditional learning techniques are limited in terms of applications of knowledge, whereas in OSS there is a clear visibility of results of skills acquisition in terms of code submitted or questions replied. This means that there is a big opportunity for transferring novel learning techniques from OSS communities to traditional learning environments [18].

One of the most obvious applications of the OSS learning approach is to use it as a part of a specific teaching module on OSS. Although learning project details normally does not require classroom participation or any other formal learning methods, it could be investigated if one could learn how to contribute to OSS within the limits of the class. Some studies provide evidence of success of such experience. One example is the Master courses in Open Source and Distributed Development Models at the University of Skövde, Sweden [16]. A group of 12 students were introduced to the OSS community aspects and required to put effort in making contributions with increasing complexity gradually overtime. As a result, the students managed to make contributions to OSS projects in terms of documentation, bug reports and desktop themes [16]. On the other hand, there are evidences that contributing to OSS can be difficult and time-consuming [12]. Therefore, it is required to keep balance between students' abilities and course requirements. Students should be evaluated in terms of their efforts or progress rather than based on the number of their code submissions or bugs fixes. This approach is adopted in the 3rd and 4th year course on Open Source Software at Nazarbayev University, Kazakhstan.

A more general exploitation of the OSS learning approach is as part of a core subject in the area of software engineering. Undergraduate students at the Aristotle University Thessaloniki, Greece, participate as project team members in real-life OSS projects as part of a course assignment on Software Engineering

[24]. Students are allowed to select an OSS project and assume a number of possible contributor roles, thus getting to understand through real participation how different professionals are involved in the software development process.

At the University of Minho, Fernandes *et al.* [9] carried out a pilot project in teaching software engineering to students in the last year of a MSc course whose completion entitles the students to teach Informatics at secondary school level. Students spontaneously got together in small groups (up to 3 elements) and chose an OSS project to get involved in. Data collection by the instructors was carried out during the pilot project using direct observation and unstructured interviews aiming at designing a learning-by-doing e-Learning framework for teaching software engineering topics.

Finally, a recent approach that may contribute to the understanding of the dynamics of learning processes is process mining [2], that is, the use of event data to extract process-related information. With the aim of understanding the learning dynamics of OSS communities, Mukala *et al.* [22] used process mining to extract learning processes from mailing archives of OSS projects. Their results provide insights on the possible discrepancies that are observed between an initial theoretical representations of learning processes and the real behavior observed from the data. Moreover, such a comparison helps foster the understanding on how learning actually takes place in OSS environments.

5 Conclusion

We have seen that OSS represents an important approach to software development and is a field that is expected to grow within the next few years. It is essential to enable newcomers to easily join OSS projects in order for the projects to be sustainable. For this reason understanding and fostering learning processes is crucial in OSS.

We discussed that learning in OSS projects can be beneficial not only to newcomers but also to core members. Newcomers can learn and acquire new skills by contributing, while core members can receive help through newcomers' contributions and possibly from their domain expertise.

We have surveyed a number of research works in the area of OSS in order to summarize how learning occurs in OSS environment. We have found that informal and project based learning methods are the most common in OSS communities. Such methods include active interaction with a community and code contributions.

We noted that the learning processes associated with such methods were identified by empirical evidences and are an essential part of the individual contributor's learning process. Finally, we discussed how such methods have been applied and can be effectively applied to education.

This paper does not pretend to provide an exhaustive review of learning methods in OSS. It essentially shows the approaches that have been investigated or have been used in the development of the course on Open Source Software at Nazarbayev University, Kazakhstan.

Part of our future work is to analyze the effectiveness of this course and its impact on the students' post-graduation involvement in OSS activities and, more in general, on their careers. Furthermore, the analysis of OSS learning carried out in previous work [9, 22] will be further developed, especially aiming to the integration of data collection through direct observation and unstructured interviews [9] and process mining techniques [22].

References

1. A. Ghosh, R., Glott, R., Krieger, B., Robles, G.: Free/libre and open source software: Survey and study (01 2002)
2. van der Aalst, W.: Process Mining - Data Science in Action. Springer, 2nd edn. (2016)
3. Androutsellis-Theotokis, S., Spinellis, D., Kechagia, M., Gousios, G.: Open source software: A survey from 10,000 feet. Foundations and Trends in Technology, Information and Operation Management **4**(3–4), 187–347 (2010)
4. Au, Y.A., Carpenter, D., Chen, X., Clark, J.G.: Virtual Organizational Learning in Open Source Software Development Projects (0013) (May 2007), https://ideas.repec.org/p/tsa/wpaper/0041is.html
5. Barcomb, A., Grottke, M., Stauffert, J.P., Riehle, D., Jahn, S.: How developers acquire floss skills. In: Damiani, E., Frati, F., Riehle, D., Wasserman, A.I. (eds.) Open Source Systems: Adoption and Impact. pp. 23–32. Springer International Publishing, Cham (2015)
6. Cerone, A.: Learning and activity patterns in oss communities and their impact on software quality. In: Proceedings of OpenCert 2011, Electronic Communications of the EASST, vol. 48 (2012)
7. Cerone, A., Sowe, S.K.: Using free/libre open source software projects as e-learning tools. In: Proceedings of OpenCert 2010, Electronic Communications of the EASST, vol. 33 (2010)
8. Edwards, K.: Epistemic communities, situated learning and open source software development. In: Proceedings from the conference on Epistemic Cultures and the Practice of Interdisciplinarity (2001)
9. Fernandes, S., Martinho, M.H., Cerone, A., Barbosa, L.S.: Integrating formal and informal learning through a floss-based innovative approach. In: Proc. of CRIWG 2013. Lecture Notes in Computer Science, vol. 8224, pp. 208–214. Springer (2013)
10. Hemetsberger, A., Reinhardt, C.: Learning and knowledge-building in open-source communities: A social-experiential approach. Management Learning **37**(2), 187–214 (2006), https://doi.org/10.1177/1350507606063442
11. Huntley, C.L.: Organizational learning in open-source software projects: an analysis of debugging data. IEEE Transactions on Engineering Management **50**(4) (2003)
12. Jaccheri, L., Osterlie, T.: Open source software: A source of possibilities for software engineering education and empirical software engineering. In: First International Workshop on Emerging Trends in FLOSS Research and Development (FLOSS'07: ICSE Workshops 2007). pp. 5–5 (May 2007). https://doi.org/10.1109/FLOSS.2007.12
13. Kim, Y., Jiang, L.: The learning curves in open-source software (oss) development network. In: Proceedings of the Sixteenth International Conference on Electronic Commerce (ICEC '14). pp. 41–48. ACM (2014), http://doi.acm.org/10.1145/2617848.2617857

14. Kuk, G.: Strategic interaction and knowledge sharing in the kde developer mailing list. Manage. Sci. **52**(7), 1031–1042 (Jul 2006), https://doi.org/10.1287/mnsc.1060.0551
15. Livingstone, D.W.: Informal learning: Conceptual distinctions and preliminary findings. Counterpoints **249**, 203–227 (2006)
16. Lundell, B., Persson, A., Lings, B.: Learning through practical involvement in the oss ecosystem: Experiences from a masters assignment. In: Feller, J., Fitzgerald, B., Scacchi, W., Sillitti, A. (eds.) Open Source Development, Adoption and Innovation. pp. 289–294. Springer US, Boston, MA (2007)
17. Marsick, V.J., Watkins, K.E.: Informal and incidental learning. New Directions for Adult and Continuing Education **89**(25-34) (2001)
18. Meiszner, A., Glott, R., Sowe, S.K.: Free / libre open source software (floss) communities as an example of successful open participatory learning ecosystems. UPGRADE, The European Journal for the Informatics Professional **9**(3), 62–68 (June 2008), http://oro.open.ac.uk/16852/
19. Meiszner, A., Glott, R., Sowe, S.K.: Preparing the ne (x) t generation: Lessons learnt from free/libre open source software why free and open are pre-conditions and not options for higher education (2008)
20. Merriam, S.B., Cafarella, R.S., Baumgartner, L.M.: Learning in adulthood : a comprehensive guide. Jossey-Bass, 3rd edn. (2007)
21. Muffatto, M.: Open Source: A Multidisciplinary Approach (Series on Technology Management). Imperial College Press, London, UK, UK (2006)
22. Mukala, P., Cerone, A., Turini, F.: An empirical verification of a-priori learning models on mailing archives in the context of online learning activities of participants in free/libre open source software (floss) communities. Education and Information Technologies **22**(6), 3207–3229 (2017)
23. Overwien, B.: Informal learning and the role of social movements. International Review of Education **46**(6), 621–640 (2000)
24. Papadopoulos, P.M., Stamelos, I.G., Meiszner, A.: Enhancing software engineering education through open source projects: Four years of students' perspectives. Education and Information Technologies **18**(2), 381–397 (2013)
25. Posnett, D., Warburg, E., Devanbu, P., Filkov, V.: Mining stack exchange: Expertise is evident from initial contributions. In: 2012 International Conference on Social Informatics. pp. 199–204 (Dec 2012). https://doi.org/10.1109/SocialInformatics.2012.67
26. Riehle, D.: The economic motivation of open source software: Stakeholder perspectives. vol. 40, pp. 25–32 (April 2007). https://doi.org/10.1109/MC.2007.147
27. Roberts, J., Hann, I., Slaughter, S.: Understanding the motivations, participation, and performance of open source software developers: a longitudinal study of the apache projects. Management Science **52**(7), 984–999 (2006)
28. Singh, P.V., Youn, N., Tan, Y.: Developer learning dynamics in open source software projects : A hidden markov model analysis (2006)
29. Sowe, S.K., Stamelos, I.: Reflection on knowledge sharing in f/oss projects. In: Russo, B., Damiani, E., Hissam, S., Lundell, B., Succi, G. (eds.) Open Source Development, Communities and Quality. pp. 351–358. Springer US, Boston, MA (2008)
30. Vonkrogh, G., Spaeth, S., Lakhani, K.: Community, joining and specialization in open source software innovation: A case study (July 2003), https://www.alexandria.unisg.ch/30623/
31. Weller, M., Meiszner, A.: Flosscom phase 2: Report on the effectiveness of a floss-like learning community in formal educational settings. FLOSSCom Project (2008)

32. Wen, S.F.: An empirical study on security knowledge sharing and learning in open source software communities. Computers **7**(4) (2018), http://www.mdpi.com/2073-431X/7/4/49
33. Xuan, Q., Filkov, V.: Building it together: Synchronous development in oss. In: Proceedings of the 36th International Conference on Software Engineering (ICSE 2014). pp. 222–233. ACM (2014), http://doi.acm.org/10.1145/2568225.2568238
34. Ye, Y., Kishida, K.: Toward an understanding of the motivation of open source software developers. pp. 419– 429 (06 2003). https://doi.org/10.1109/ICSE.2003.1201220