The Hard Life of Open Source Software Project Newcomers

Igor Steinmacher, Igor Scaliante Wiese
DACOM – UTFPR
Campo Mourão-PR - Brazil
{igorfs, igor}@utfpr.edu.br

Tayana Conte
IComp – UFAM
Manaus-AM - Brazil
tayana@icomp.ufam.edu.br

Marco Aurelio Gerosa
IME – USP
São Paulo-SP - Brazil
gerosa@ime.usp.br

David Redmiles
Department of Informatics
Univ. of California, Irvine
Irvine-CA - USA
redmiles@ics.uci.edu

ABSTRACT
While onboarding an open source software (OSS) project, contributors face many different barriers that hinder their contribution, leading in many cases to dropouts. Many projects leverage the contribution of outsiders and the sustainability of the project relies on retaining some of these newcomers. In this paper, we discuss some barriers faced by newcomers to OSS. The barriers were identified using a qualitative analysis on data obtained from newcomers and members of OSS projects. We organize the results in a conceptual model composed of 38 barriers, grouped into seven different categories. These barriers may motivate new studies and the development of appropriate tooling to better support the onboarding of new developers.

Categories and Subject Descriptors

D.2.9 **[Management]:** Programming teams
K.4.3 [**Organizational Impacts**]

General Terms

Human Factors.

Keywords
Open Source Software, newcomers, newbies, new developers, barrier, joining, onboarding, qualitative study, Grounded Theory.

1. INTRODUCTION
A continuous influx of newcomers and their active engagement in development activities are crucial to the success of Open Source Software (OSS) projects [26]. Therefore, a major challenge for OSS projects is to provide ways to support the joining of newcomers.

We claim that joining a project is a complex process composed of different stages and a set of forces that push newcomers towards or away from the project. We consider that four different forces influence this process: *newcomers' motivation*, *project attractiveness, onboarding barriers* and *retention* [33]. *Motivation* and project *attractiveness* are the forces that draw the outsider to contribute to a project. While motivation persists as an ongoing force, various *barriers* and *retention* forces influence onboarding, contribution, and members' permanence [33]. Understanding developer motivation and project attractiveness are well-explored topics in the literature [12, 18, 20, 23, 28, 32, 42]. However, little is known about the barriers that newcomers face when onboarding a project, a process that still presents many open issues [41].

The onboarding stage is highly impacted by a steep learning curve as well as reception and expectation breakdowns, longer-term forces influence the contributing stage. Moreover, not every developer wants to become a contributor, committer, or a core member, although all of them are subject to the problems of onboarding before making their first contribution.

Dagenais et al. [9], for example, compare software project newcomers to explorers who need to orient themselves in a hostile environment. On the one hand, newcomers need to learn social and technical aspects alone, exploiting existing information in mailing lists, source code repositories, and issue managers [29]. On the other hand, it is not easy to access this information due to the large volume, lack of tools to navigate the repository, and the difficulty of linking logically related items in different sources [8].

OSS projects can benefit from more contributions if they offer the right support to newcomers during their onboarding. To achieve this, it is necessary to understand how OSS communities interact and what barriers are for newcomers to OSS projects when they are onboarding such projects.

In this paper, we present a preliminary discussion about the barriers hindering newcomers' onboarding to OSS projects. The barriers were identified using a qualitative analysis on data obtained from two different sources: feedback from 9 graduate and undergraduate students; and 24 responses to an open-ended questionnaire sent to OSS communities. The analysis resulted in a model of barriers that can be used to motivate future research and to serve as requirements for building tools to support OSS projects and their newcomers.

Therefore, the contribution of this paper is two-fold. First, we present a model of barriers based on data collected from newcomers and OSS developers. Second, it brings to light the challenges associated to these barriers , which demand studies from different research areas. This paper is the starting point to model and organize the problems that occur in practice, but are not reported in the literature.

2. RELATED WORK
Newcomers' onboarding is not an issue exclusively faced by OSS. Many studies in the literature deal with newcomers joining process in collective production communities, including studies on Wikipedia [11, 39] and on open software projects [4, 13, 19, 21, 35, 40]. Dagenais et al. [9] and Begel and Simon [2] present studies regarding newcomers joining process in software projects, but their focus is in industrial settings.

Von Krogh et al. [19] analyzed interviews with developers, emails, source code repository, and documents of the FreeNet project. The authors proposed a joining script for developers who

CHASE'14, June 2 – June 3, 2014, Hyderabad, India
Copyright 2014 ACM 978-1-4503-2860-9/14/06...$15.00
http://dx.doi.org/10.1145/2593702.2593704

want to take part in the project. Nakakoji et al. [25] studied four OSS projects to analyze the evolution of these communities. They presented eight possible roles for the community members and structured them into a model composed of concentric layers, like the layers of an onion. This structure was later called the onion patch, and other authors conducted studies based on this model [13, 15, 16]. Although these papers deal with the evolution of members' participation in OSS communities, they focus on newcomers after the onboarding.

Some researchers tried to understand the barriers that influence the retention of newcomers are. Zhou and Mockus [44] worked on identifying the newcomers who are more likely to remain in the project in order to offer active support for them to become long-term contributors. Jensen et al. [13] analyzed mailing lists of OSS projects to verify if the emails sent by newcomers are quickly answered, if gender and nationality influence the kind of answer received, and if the reception of newcomers is different in users and developers lists. Steinmacher et al. [35] used data from mailing list and issue tracker to study how reception influences the retention of newcomers in an OSS project.

There are also some studies presenting tools to support newcomers' first steps. Čubranić et al. [8] presented Hipikat, a tool that supports newcomers by building a group memory and recommending source code, mails messages, and bug reports to support newcomers. Wang and Sarma [40] present a Tesseract extension to enable newcomers to identify bugs of interest, resources related to that bug, and visually explore the appropriate socio-technical dependencies for a bug in an interactive manner. Park and Jensen [26] show that visualization tools support the first steps of newcomers in an OSS project, helping them to find information more quickly.

Mentoring is also explored as a way to support newcomers. Malheiros et al. [21] and Canfora et al.[4] proposed different approaches to identify and recommend mentors to newcomers of OSS projects by mining data from mailing lists and source code versioning systems.

As listed, there are some efforts to study newcomers to OSS. Moreover, we conducted a systematic literature review on barriers faced by newcomers to OSS [34] and, for the best of our knowledge, there are no studies focusing specifically on identifying the barriers for newcomers to OSS projects. In this paper, we try to identify these barriers and provide motivation for further researches related to them.

3. IDENTIFYING ONBOARDING BARRIERS

To understand the key barriers for newcomers during onboarding, we performed a qualitative analysis on feedback obtained from students and from members of different OSS projects.

3.1 Data Collection

We conducted a qualitative analysis over data obtained from newcomers and from community members. Newcomers' data consists of nine feedback received from PhD candidates and undergraduate students after contributing to OSS projects as part of a course assignment. All the students were newcomers to the projects they were contributing. The PhD candidates were all males, experienced developers, with 30 years old or more. The undergraduate students were four males and one female, with ages among 21 and 24 year old, and were attending to the last semester of Internet Systems course, therefore, about to join the software development industry.

The students contributed to the projects JabRef (2 graduate/2 undergraduate), LibreOffice (2 undergraduate), Mozilla Firefox (3 graduate). After accomplishing the assignment, their feedback was collected by means of an open-ended questionnaire. We created the questionnaire using LimeSurvey[1] and the students answered it via internet. The goal of the questions was to enable students to debrief, and provide the general problems they faced during their onboarding.

The community data was obtained from 24 answers to an open question sent to developers mailing lists and forums of OSS projects. The messages were posted and the answers received during October, 2013. We sent the message to 6 different projects: atunes, audacity, LibreOffice, Apache OpenOffice, Mozilla Firefox, and jEdit. We chose projects from different business domains. It is important to notice that none of them deliver development frameworks or scaffolding technologies, since this kind of project usually is generally more complex and demand higher and more specific skills and knowledge. These characteristics could hide some possible barriers encountered by newcomers, once these newcomers can face complex problems related to technological and learning gap during onboarding.

The questionnaire delivered to the community members was composed of two questions to profile the contributor (project and contribution time), and an open question: "***In your opinion, what are the main difficulties faced by newcomers when they want to start contributing to this project? (Consider technical and non-technical issues)*.**"

We received 24 complete answers to the questionnaire, from contributors of eight different projects, as presented in Table 1. Regarding how long they had been contributing to the project, the distribution is presented in Table 2. We received answers from people that contributed to 6 different projects, and that contributed to the projects for different periods (ranging from newcomers to experienced members).

Table 1. Project to which participants mainly contribute

Project	Count	Percentage
LibreOffice / Apache OpenOffice	9	37.50%
aTunes	3	12.50%
Mozilla Firefox	3	12.50%
Audacity	2	8.33%
jEdit	1	4.17%
OpenVPN	1	4.17%
FreePlane	1	4.17%
Emacs	1	4.17%
Did not inform	3	12.50%

Table 2. Period of contribution for questionnaire respondents

For how long have you being contributing to the project?	Count	Percentage
Less than 6 months	7	29.17%
Between 6 months and 1 year	3	12.50%
Between 1 year and 3 years	6	25.00%
More than 3 years	8	33.33%

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

3.2 Qualitative Data Analysis

We analyzed the data using procedures of Grounded Theory (GT) [38], which is based in the concept of coding. Coding can be divided into three steps: open coding, where concepts are identified and their properties and dimensions are discovered in the data; axial coding, where connections between the codes are identified and grouped according to their properties to represent categories; and selective coding, where the core category (that integrates the theory) is identified and described. In this study, we applied just the open and axial coding steps, because it allows us to identify the barriers for newcomers to OSS.

The open coding process was conducted in parallel by three researchers. Each researcher quoted and coded the documents independently. After coding, we discussed the quotes and codes until they come to a consensus for the whole set of documents. After the discussion, we started some iterations of axial coding, followed by discussions and changes in codes and categories. In the next section, we present the results from this analysis.

3.3 Results and Discussion

During the coding process, seven categories emerged from the data along with 38 barriers. Table 3 presents an overview of these categories, as well as the count of the documents, quotes, and barriers coded. The count of documents is also reported in terms of count of feedback and count of answers to the open question in which that category appeared.

The identified categories represent a group of barriers that the participants believed to affect the participation of newcomers to OSS projects. Following, we provide a brief description of these categories, highlighting some of the barriers evidenced.

Table 3. Overview of Categories that Emerged

Category	# of documents (feedback/question)	#quotes	# barriers
Issues to build/set up workspace	8 (4 / 4)	15 (10/5)	5
Code issue	15 (7 / 8)	21 (11/10)	5
Problem with documentation	15 (8 / 7)	23 (15/8)	10
Newcomer Behavior	3 (0 / 3)	3 (0/3)	2
Newcomer Tech. Knowledge	12 (4 / 8)	16 (7/9)	7
Social Interaction Issue	11 (6 / 5)	12 (8/4)	6
Finding a way to start	11 (8 / 3)	22 (18/4)	3

Issues to build/set up the workspace: setting up and building the workspace hinder newcomer onboarding to OSS. To modify the application it is necessary to build the application locally first, what can take time and demotivate the newcomer. During our analysis, this category appeared in eight documents, and is related to the following barriers: issues setting up workspace, platform dependency, problem finding out the correct source code, and library dependencies. One participant reported that "*the biggest problem was how to get project from SCM and it to work properly."* This quote is illustrates the barrier issues setting up workspace, the most reported barrier (eight quotes in six documents) under this category.

Table 4 presents the barriers and who reported them split according to source and period of contribution of the participant. We can see that the more experienced members do not see this category of barriers as a real issue. However, students and new developers reported their issues. Thus, in this case, it is necessary to make the community aware of this issue so they have a chance to provide the right support to the newcomers.

There are many open research opportunities and challenges related to building/setting up workspace barriers. For example: what is the most effective solution to this problem from the newcomers' perspective? What tooling is needed to make it simple for the community to offer these solutions? Can mechanisms to support newcomers building the workspace increase the number of contributors/contributions or is this barrier necessary to filter interested/skilled newcomers and keep the quality of code?

Table 4. Build/set up workspace barriers per data source and time in the project

Data Source	Feedback	Questions		
Background/ Time in the project	Students	Less than 6 months	Between 6 months and 3 years	More than 3 years
Issues setting up	•	•	•	
Platform dependency	•		•	
Finding the correct source code	•			
Library dependencies	•			

Code issues: this category comprises the barriers that are related to the source code of the products. To contribute a newcomer usually needs to change existing source code. Therefore, it is necessary for the newcomer to have enough knowledge about the code to start his contributions. Figure 1 presents the barriers related to this category: codebase size, bad quality of code, lack of code standards, problem to understand the source code, and outdated code. Regarding problems understanding the source code (identified in five documents), one participant reported that "*the main difficulty was getting used to the code*" and pointed as a possible cause the need to "*define very clearly what the standards of the developed software, including the class and methods naming.*" Codebase size is also frequently cited, and can be summarized by this quote: "*huge codebase that takes time to learn.*" Bad quality of code also deserves to be evidenced. The following quotes (from respondents) illustrate this barrier: "*Relatively poor code quality*"; a problem is "*the junk code.*"

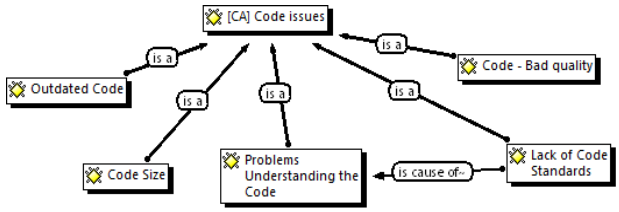

Figure 1. Graphical representation with the associations related to the Code Issues barriers category

In **Error! Not a valid bookmark self-reference.**, we present the barriers and who reported them split according to the data source and the period of contribution of the participant. Here we can see that newcomer students do not see the codebase size as an issue, while practitioners mentioned it as a possible barrier. Possibly, the newcomers reported the same barrier as a problem to understand the code, which was largely mentioned by the students. It is also possible to verify that the quality of code is reported by both students and long term contributors.

Table 5. Code issues reported per data source and time in the project

Data Source / Background/ Time in the project	Feedback Students	Questions: Less than 6 months	Questions: Between 6 months and 3 years	Questions: More than 3 years
Bad Quality of Code	•		•	•
Codebase Size		•	•	•
Outdated Code		•		
Problems Understanding the code	•	•	•	
Lack of Code Standards	•			

Strategies to improve code comprehension and bad quality code detection are topics already tackled in current literature [1, 6, 17, 31]. However, we are not aware of the use of techniques and methods proposed in the literature to support newcomers to OSS. More studies to verify the effectiveness of using these approaches with OSS newcomers is a fruitful area of research. An example of study is the one conducted by Park and Jensen [26], who analyzed the effectiveness of visualization tools to support newcomers understanding about the code architecture.

Problems with documentation: newcomers need to learn technical and social aspects of the project to contribute. Thus, problems related to documentation were recurrently reported. In our analysis, we identified ten barriers under this category, as depicted in Figure 2. A barrier that deserves attention here is the lack of documentation. This barrier appears in four documents (five quotes), and it is reported partially as the lack of six specific types of documentation. The following quotes illustrate some of these types:

"*There is no clear documentation about the project organization…*" (lack of documentation on project structure)

"*Knowing the exact process of contributing (to any given project), which may or may not be documented accurately…*" (lack of contribution process documentation)

"*…because there was no documentation on how to run the project, or on its dependencies"* (lack of documentation on setting up workspace)Table 6 Other than lack of documentation, we also identified spread documentation, unclear documentation, and outdated documentation as barriers for newcomers. On the other hand, some people reported positive newcomers about documentation, which contradicts four barriers. For example, a respondent of the open question told that "there's tons of up-to-date documentation;" and a student reported that "the project provided all the information for building the project."

Table 6 reports the barriers and who reported them. We can see that, again, the more experienced members did not perceive the issues reported by the newcomers. When looking at students' feedback, we see that six documentation issues were reported, showing how these barriers can hinder the first steps of newcomers.

Lack and improper documentation lead to problems that affect not only OSS projects, but also software projects in general. Understanding the code structure and the application architecture with a proper documentation is harder. However, it is interesting to investigate why documentation is perceived as a barrier by the newcomers: Is there a lack of documentation or the problem is how to find the proper documentation? Is it easy to reach the right

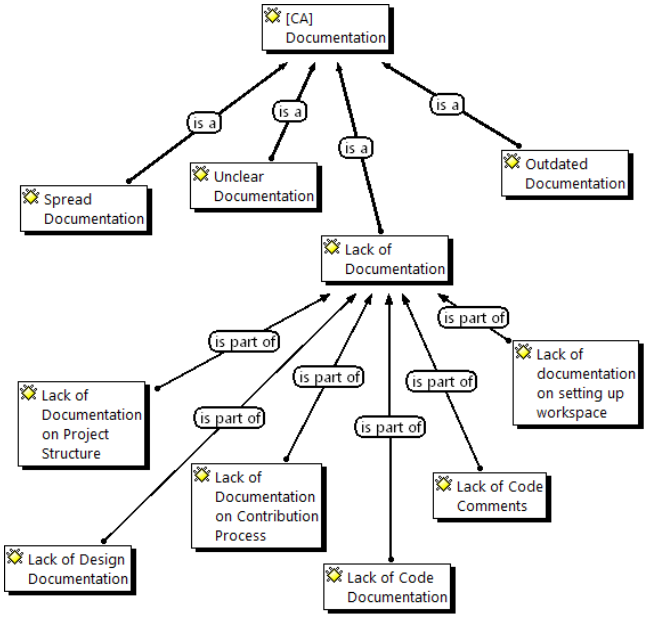

Figure 2. Graphical representation of the associations related to the Documentation Issues barriers category

documentation?

Other than lack of documentation, we also identified spread documentation, unclear documentation, and outdated documentation as barriers for newcomers. On the other hand, some people reported positive newcomers about documentation, which contradicts four barriers. For example, a respondent of the open question told that "*there's tons of up-to-date documentation;*" and a student reported that "*the project provided all the information for building the project*."

Table 6. Problems with documentation per data source and time in the project

Data Source / Background/ Time in the project	Feedback Students	Questions: Less than 6 months	Questions: Between 6 months and 3 years	Questions: More than 3 years
Lack of Documentation	•	•	•	
Lack of Documentation on Proj. Structure	•			
Lack of Documentation on setting up workspace	•			
Lack of Documentation on Contribution Process	•			•
Outdated documentation	•		•	
Unclear documentation	•			
Spread documentation			•	
Lack of Code Comments		•		
Lack of Design Documentation		•		
Lack of Code Documentation			•	

Newcomer Behavior: two barriers were identified under this category: lack of newcomer commitment and underestimating the challenge. One member reported that newcomers "*often underestimate the challenge*." Another reported that a problem is that newcomers need "*courage to engage with the development community*." These barriers were raised by three respondents, all of them members of community, as presented in Table 7.

Table 7. Newcomers' behavior barriers organized by source and profile

Data Source / Background/ Time in the project	Feedback Students	Questions: Less than 6 months	Questions: Between 6 months and 3 years	Questions: More than 3 years
Lack of Commitment		•		•
Underestimating the challenge				•

Newcomer Technical Knowledge: seven barriers related to newcomer knowledge were reported in 12 documents analyzed. The barriers identified include previous knowledge on project tooling, choosing the right tooling, previous knowledge on versioning control systems (VCS), lack of knowledge on technologies used, programming language used, learning curve, and learning curve of project tooling. Regarding lack of knowledge on technologies, a student reported: "*my previous knowledge was very little ... as the projects use different frameworks, you need to understand these frameworks in order to contribute.*" An open question respondent reported "*understanding obscure old C++*" as a difficulty that evidences programming language knowledge. Learning curve to use the project tooling appears in three different documents, as illustrated by the quote: "*To become acquainted with the used tooling, when the project is rather new and/or changed the tooling it used.*" This category shows that newcomers that wish to contribute must check if the technical skills required for a given task or project match with their skills.

Both newcomers and community members recognize previous knowledge as a barrier that hinders newcomers' onboarding, as it can be observed in Table 8. The point to investigate is how to make the newcomers aware of the skills needed; and, how to support them choosing the right tasks that fits with their knowledge? This is in line with the barriers presented in the next category, "find a way to start."

Table 8. Newcomers' previous knowledge barriers per data source and time in the project

Data Source / Background/ Time in the project	Feedback Students	Questions: Less than 6 months	Questions: Between 6 months and 3 years	Questions: More than 3 years
Previous knowledge on project tooling	•		•	
Previous knowledge on VCS		•		•
Choosing the right tooling		•		
Lack of knowledge on technologies used	•			
Programming language used	•			
Learning curve			•	
Learning curve on project tooling			•	•
General lack of knowledge	•		•	•

Finding a way to start: this category represents the barriers related to difficulties that newcomers face when trying to find the right place to start contributing. One of these barriers, find a task to start, is the barrier that appeared in the highest number of documents analyzed, nine. A quote from a student feedback makes it clear: "*We do not know what is easy when we join a project, or at least the size of the problem that we are getting into. It is necessary to take a risk and try a few possibilities.*" This barrier is also supported by the answers received from the community, for example: "*it's not always clear where someone new can jump in and make an impact.*" There are other two barriers identified in this category: find the right piece of code to work on ("*I don't know what are the easiest ones and what part of code should I start looking at*") and outdated list of bugs ("*Issues on bug tracker are not closed frequently*").

In Table 9 we present the evidenced barriers according to the data source and profile of the reporter. We can see that students that were onboarding the project largely reported barriers that are under this category.

Table 9. Finding a way to start barriers quotes organized by data source and time in the project

Data Source / Background/ Time in the project	Feedback Students	Questions: Less than 6 months	Questions: Between 6 months and 3 years	Questions: More than 3 years
Find the right piece of code to work	•	•		
Outdated list of bugs	•			
Find a task to start	•	•		•

Regarding finding the right piece of code to work, we found some studies in the literature that can support the newcomers [7, 22]. Although, the selection of the most appropriate task developers should choose deserves further investigation, and can be a fruitful topic. A possible research question to be addressed is: How to recommend first tasks to newcomers based on their skills or goals?

Social Interactions Issues: this category comprises the barriers related to interaction between newcomer and the community. Among these barriers, we highlight finding someone to help. A newcomer reported that "*there should be someone responsible for receiving and coordinating the onboard of new members in the project*," while a respondent of the open question said that "*It is hard to get someone to give us this kind of information* [*find where to start*]." Finding mentor was also identified as part of the barrier finding someone to help. This category also comprises barriers related to communication issues on the mailing list, including: impolite answers ("*at the beginning it seemed that they did not want help*"); use of intimidating terms and abbreviations; and delayed responses ("*it took time to receive answers to our email*"). However, these barriers related to mailing lists were contradicted by some other respondents, that highlighted the good support offered by the community ("*I was surprised at the way people answered, always very courteous…*") and the prompt answers on the mailing list ("*there's an active development mailing list where newcomers can ask questions which are usually promptly answered*").

In Table 10 we can observe that social issues are reported by the students and by community members that joined the projects recently. In students' case, a possible explanation is that they needed prompt support, as they had a tight schedule to accomplish the assignment. The barriers that are related to finding a mentor/help, have been studied in some researchers, and some tools were already proposed [5, 24, 30, 37].

Table 10. Social Interaction barriers quotes per data source and time in the project

Data Source / Background/ Time in the project	Feedback Students	Questions: Less than 6 months	Questions: Between 6 months and 3 years	Questions: More than 3 years
Delayed Responses	•			
Impolite answers	•			
Finding someone to help	•	•		
Use of intimidating terms		•		
Communication issues		•		
Finding a mentor				•

In fact, social interactions during newcomers' onboarding have been widely studied lately. They usually analyze historical data from the projects to understand how socialization of newcomers [3, 10, 27], project receptiveness[14, 36] and project social environment [43, 45] influence newcomers onboarding.

4. CONCLUSION AND FUTURE WORK

The results of our qualitative analysis based on feedback from newcomers and members of OSS projects resulted in the emergence of seven categories of barriers that hinder newcomer onboarding to OSS projects.

The qualitative analysis helped us to identify barriers and relationships of barriers that influence the newcomer onboarding in OSS projects. It is possible to notice that the onboarding process of a newcomer to an OSS can be a tough task. Each of the categories presented can motivate and provide insights to further researches and enable investigations in different perspectives.

In the particular case of this study, we found that the two most reported barriers - find a task to start and problems setting up the local workspace - are not well explored by the literature.

We believe that by using these results it is possible to offer the insights needed to start researching ways to facilitate the influx of newcomers to OSS projects. We found barriers that can motivate researches in different areas, including behavioral sciences, collaborative systems, human computer interactions, and their intersections with software engineering.

We recognize that the sample used was not the ideal, comprising two different sources of data and involving subjects from a many different projects and different skill levels. This is justified once we are focusing on finding barriers that are common to different OSS projects. Moreover, this study is just a preliminary result of a broader research that aims at investigating and understanding the barriers faced by newcomers in OSS projects.

To further investigate these barriers we are conducting a series of interviews with core members and newcomers identified in OSS projects to confirm the findings of the results of our qualitative studies. In addition to the challenges presented, future directions include surveying a larger sample of practitioners to generalize the qualitative findings. Moreover, we aim building a dashboard to improve newcomers' awareness on project related information. We also plan to map the barriers found to tools proposed in the state-of-art and to those available in the state-of-practice.

5. ACKNOWLEDGMENTS

The authors thank Fundação Araucária for the financial support. Marco receives individual grant from the CNPq and FAPESP. Igor receives grant from CAPES (BEX 2038-13-7). Igor Wiese receives grants from CAPES (BEX 2039-13-3). David Redmiles receives a grant from the US National Science Foundation (#1111446).

6. REFERENCES

[1] Bassil, S. and Keller, R.K. 2001. Software visualization tools: survey and analysis. *Program Comprehension, 2001. IWPC 2001. Proceedings. 9th International Workshop on* (2001), 7–17.

[2] Begel, A. and Simon, B. 2008. Novice Software Developers, All over Again. *4th Intl. Workshop on Computing Education Research* (2008), 3–14.

[3] Bird, C. 2011. Sociotechnical coordination and collaboration in open source software. *Proceedings of the 2011 27th IEEE International Conference on Software Maintenance* (Washington, DC, USA, 2011), 568–573.

[4] Canfora, G., Di Penta, M., Oliveto, R. and Panichella, S. 2012. Who is Going to Mentor Newcomers in Open Source Projects? *20th Intl. Symposium on the Foundations of Software Engineering* (2012), 44:1–44:11.

[5] Canfora, G., Di Penta, M., Oliveto, R. and Panichella, S. 2012. Who is Going to Mentor Newcomers in Open Source Projects? *Proceedings of the ACM SIGSOFT 20th International Symposium on the Foundations of Software Engineering* (Cary, North Carolina, 2012), 44:1–44:11.

[6] Cornelissen, B., Zaidman, A., Deursen, A. van, Moonen, L. and Koschke, R. 2009. A Systematic Survey of Program Comprehension through Dynamic Analysis. *Software Engineering, IEEE Transactions on*. 35, 5 (2009), 684–702.

[7] Cubranic, D. and Murphy, G.C. 2003. Hipikat: recommending pertinent software development artifacts. *Software Engineering, 2003. Proceedings. 25th International Conference on* (May. 2003), 408–418.

[8] Cubranic, D., Murphy, G.C., Singer, J. and Booth, K.S. 2005. Hipikat: a project memory for software development. *IEEE Transactions on Software Engineering*. 31, (2005), 446–465.

[9] Dagenais, B., Ossher, H., Bellamy, R.K.E., Robillard, M.P. and Vries, J.P. de 2010. Moving into a New Software Project Landscape. *Proceedings of the 32sd ACM/IEEE Intl. Conf. on Software Engineering* (2010), 275–284.

[10] Ducheneaut, N. 2005. Socialization in an Open Source Software Community: A Socio-Technical Analysis. *Computer Supported Cooperative Work (CSCW)*. 14, 4 (2005), 323–368.

[11] Halfaker, A., Kittur, A. and Riedl, J. 2011. Don't Bite the Newbies: How Reverts Affect the Quantity and Quality of Wikipedia Work. *7th Intl. Symposium on Wikis and Open Collaboration* (2011), 163–172.

[12] Hars, A. and Ou, S. 2001. Working for free? Motivations of participating in open source projects. *34th Annual Hawaii Intl. Conference on System Sciences* (2001), 9 pp.

[13] Jensen, C., King, S. and Kuechler, V. 2011. Joining Free/Open Source Software Communities: An Analysis of Newbies' First Interactions on Project Mailing Lists. *44th Hawaii Intl. Conf. on System Sciences* (2011), 1–10.

[14] Jensen, C., King, S. and Kuechler, V. 2011. Joining Free/Open Source Software Communities: An Analysis of Newbies' First Interactions on Project Mailing Lists. *System Sciences (HICSS), 2011 44th Hawaii International Conference on* (2011), 1–10.

[15] Jensen, C. and Scacchi, W. 2007. Role Migration and Advancement Processes in OSSD Projects: A Comparative Case Study. *29th Intl. Conf. on Software Engineering, 2007. ICSE 2007.* (2007), 364–374.

[16] Jergensen, C., Sarma, A. and Wagstrom, P. 2011. The Onion Patch: Migration in Open Source Ecosystems. *19th ACM SIGSOFT Symposium and the 13th European Conf. on Foundations of Software Engineering* (2011), 70–80.

[17] Khomh, F., Vaucher, S., Gueheneuc, Y.-G. and Sahraoui, H. 2009. A Bayesian Approach for the Detection of Code and Design Smells. *Quality Software, 2009. QSIC '09. 9th International Conference on* (Aug. 2009), 305–314.

[18] Krishnamurthy, S. 2006. On the intrinsic and extrinsic motivation of free/libre/open source (FLOSS) developers. *Knowledge, Technology & Policy*. 18, 4 (2006), 17–39.

[19] Krogh, G. von, Spaeth, S. and Lakhani, K.R. 2003. Community, joining, and specialization in open source software innovation: a case study. *Research Policy*. 32, (2003), 1217–1241.

[20] Lakhani, K.R. and Wolf, R.G. 2005. Why Hackers Do What They Do: Understanding Motivation and Effort in Free/Open Source Software Projects. *Perspectives on Free and Open Source Software*. MIT Press.

[21] Malheiros, Y., Moraes, A., Trindade, C. and Meira, S. 2012. A Source Code Recommender System to Support Newcomers. *36th Annual Computer Software and Applications Conf. (COMPSAC)* (2012), 19–24.

[22] Malheiros, Y., Moraes, A., Trindade, C. and Meira, S. 2012. A Source Code Recommender System to Support Newcomers. *36th Annual Computer Software and Applications Conf. (COMPSAC)* (2012), 19–24.

[23] Meirelles, P., Santos, C., Miranda, J., Kon, F., Terceiro, A. and Chavez, C. 2010. A study of the relationships between source code metrics and attractiveness in free software projects. *Software Engineering (SBES), 2010 Brazilian Symposium on* (2010), 11–20.

[24] Mockus, A. and Herbsleb, J.D. 2002. Expertise browser: a quantitative approach to identifying expertise. *Proceedings of the 24th International Conference on Software Engineering* (Orlando, Florida, 2002), 503–512.

[25] Nakakoji, K., Yamamoto, Y., Nishinaka, Y., Kishida, K. and Ye, Y. 2002. Evolution Patterns of Open-source Software Systems and Communities. *Intl. Workshop on Principles of Software Evolution* (2002), 76–85.

[26] Park, Y. and Jensen, C. 2009. Beyond pretty pictures: Examining the benefits of code visualization for Open Source newcomers. *5th Intl. Workshop on Visualizing Software for Understanding and Analysis* (2009), 3–10.

[27] Qureshi, I. and Fang, Y. 2011. Socialization in Open Source Software Projects: A Growth Mixture Modeling Approach. *Org. Res. Methods*. 14, 1 (2011), 208–238.

[28] Santos, C., Kuk, G., Kon, F. and Pearson, J. 2013. The Attraction of Contributors in Free and Open Source Software Projects. *J. Strateg. Inf. Syst.* 22, 1 (Mar. 2013), 26–45.

[29] Scacchi, W. 2002. Understanding the requirements for developing open source software systems. *Software, IEE Proceedings* -. 149, (2002), 24–39.

[30] Schuler, D. and Zimmermann, T. 2008. Mining usage expertise from version archives. *Proceedings of the 2008 international working conference on Mining software repositories* (Leipzig, Germany, 2008), 121–124.

[31] Schumacher, J., Zazworka, N., Shull, F., Seaman, C. and Shaw, M. 2010. Building Empirical Support for Automated Code Smell Detection. *Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement* (Bolzano-Bozen, Italy, 2010), 8:1–8:10.

[32] Shah, S.K. 2006. Motivation, Governance, and the Viability of Hybrid Forms in Open Source Software Development. *Manage. Sci.* 52, 7 (2006), 1000–1014.

[33] Steinmacher, I., Gerosa, M.A. and Redmiles, D. 2014. Attracting, Onboarding, and Retaining Newcomer Developers in Open Source Software Projects. *Workshop on Global Software Development in a CSCW Perspective* (2014).

[34] Steinmacher, I., Silva, M.A.G. and Gerosa, M.A. 2014. Systematic review on problems faced by newcomers to open source projects. *10th International Conference on Open Source Software* (2014), 10pp.

[35] Steinmacher, I., Wiese, I., Chaves, A.P. and Gerosa, M.A. 2013. Why do newcomers abandon open source software projects? *6th Intl. Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)* (2013), 25–32.

[36] Steinmacher, I., Wiese, I.S., Chaves, A.P. and Gerosa, M.A. 2012. Newcomers Withdrawal in Open Source Software Projects: Analysis of Hadoop Common Project. *SBSC* (2012), 65–74.

[37] Steinmacher, I., Wiese, I.S. and Gerosa, M.A. 2012. Recommending mentors to software project newcomers. *Recommendation Systems for Software Engineering (RSSE), 2012 Third International Workshop on* (Washington, DC, USA, Jun. 2012), 63–67.

[38] Strauss, A. and Corbin, J.M. 1998. *Basics of Qualitative Research : Techniques and Procedures for Developing Grounded Theory*. SAGE Publications.

[39] Vora, P., Komura, N. and Team, S.U. 2010. The N00B Wikipedia Editing Experience. *6th Intl. Symposium on Wikis and Open Collaboration* (2010), 36:1–36:3.

[40] Wang, J. and Sarma, A. 2011. Which Bug Should I Fix: Helping New Developers Onboard a New Project. *4th Intl. Workshop on Cooperative and Human Aspects of Software Engineering* (2011), 76–79.

[41] Wolff-Marting, V., Hannebauer, C. and Gruhn, V. 2013. Patterns for tearing down contribution barriers to FLOSS projects. *12th Intl. Conf. on Intelligent Software Methodologies, Tools and Techniques* (2013), 9–14.

[42] Ye, Y. and Kishida, K. 2003. Toward an Understanding of the Motivation Open Source Software Developers. *Proceedings of the 25th International Conference on Software Engineering* (Portland, Oregon, 2003), 419–429.

[43] Zhou, M. and Mockus, A. 2011. Does the initial environment impact the future of developers. *Software Engineering (ICSE), 2011 33rd International Conference on* (May. 2011), 271–280.

[44] Zhou, M. and Mockus, A. 2012. What Make Long Term Contributors: Willingness and Opportunity in OSS Community. *2012 Intl. Conf. on Software Engineering* (2012), 518–528.

[45] Zhou, M. and Mockus, A. 2012. What make long term contributors: Willingness and opportunity in OSS ommunity. *Software Engineering (ICSE), 2012 34th International Conference on* (Jun. 2012), 518–528.