Preliminary empirical identification of barriers faced by newcomers to Open Source Software projects

Igor Steinmacher, Ana Paula Chaves
DACOM – UTFPR
Campo Mourão, PR – Brazil
{igorfs, anachaves}utfpr.edu.br

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

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

***Abstract*— When newcomers try to join an open source software (OSS) project, they face many barriers that hinder their first contribution, leading in many cases to their dropping out. Many projects leverage the contribution of outsiders, and the sustainability of the project relies on retaining some of these newcomers. This research aims to identify the barriers that hinder newcomers' onboarding to OSS projects. Our method consisted of a qualitative study conducted with data obtained from four different sources: (i) systematic literature review; (ii) feedback from nine graduate and undergraduate students after they tried to join OSS projects; (iii) 24 responses to a questionnaire sent to 9 OSS projects; and (iv) semi-structured interviews with 36 subjects from 14 different projects, including newcomers and experienced members. The method to select the candidate papers in the systematic literature review was querying four digital libraries and backward snowballing. The data obtained from the practitioners from all three sources, and the primary studies obtained in the systematic review were analyzed using used procedures of Grounded Theory's open and axial coding. The analysis resulted in a conceptual model composed of 58 barriers, grouped into six different categories: cultural differences, newcomers' characteristics, reception issues, orientation, technical hurdles, and documentation problems. We could observe recurrent barriers evidenced in different data sources. We could notice that the onboarding process of a newcomer to an OSS can be a tough task. This research brings empirical support relying on data from different sources, organizes and discusses the existing common wisdom about barriers faced by newcomers to OSS projects, which deserve attention from researchers and OSS communities.**

Keywords—newcomers, onboarding, open source software, qualitative analysis, systematic literature review

I. Introduction

Many Open Source Software (OSS) projects leverage contributions from distributed volunteers. A continuous influx of newcomers is necessary for their survival, long-term success, and continuity. According to Qureshi and Fang [1], it is essential to motivate, engage, and retain new developers in order to promote a sustainable community in a project. Some studies report that newcomers are a source of innovation, new ideas, and work procedures [2].

However, newcomers usually face many difficulties to make their first contribution to an open source project. Dagenais et al. [3] compare software project newcomers to explorers who need to orient themselves in a hostile environment. In OSS projects, newcomers are usually left to learn on their own [4]. A major challenge for OSS projects is to provide ways to support newcomers' joining.

We claim that joining a project is a complex process composed of different stages influenced by forces that push newcomers towards or away from the project. We split the joining process into two different stages: *onboarding* and *contributing*, since there are different emphases in each one of them. While 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 [5], although all of them are subject to the problems of onboarding before making their first contribution.

In a previous work [6], we defined a "developer joining model" representing the stages that are common to and the forces that are influential to newcomers being drawn or pushed away from a project. We consider that joining an OSS project is a process influenced by four different forces. *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 [6]. Understanding developer motivation and project attractiveness are well-explored topics in the literature [7–10]. However, little is known about the barriers that newcomers face when onboarding a project, a process that still presents open issues [11].

When developers decide to support an OSS project, they need to learn social and technical aspects of the project before placing a contribution. During this learning period, newcomers face barriers that can result in their decision to give up contributing. Karl Fogel, in his book [12], states that *"if a project doesn't make a good first impression, newcomers may wait a long time before giving it a second chance."* 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 what barriers affect newcomers to OSS projects during their onboarding.

The goal of our research was to identify and organize the barriers for newcomers' onboarding to OSS projects. To provide a broader set of barriers, we conducted a qualitative study relying on four different sources: (i) a systematic literature review (SLR) aimed at identifying and organizing the barriers evidenced by the literature [13]; (ii) the feedback from 9 students after they contributed to OSS projects; (iii) 24 answers to an open-question sent to 9 OSS projects; and (iv) semi-structured interviews with 36 developers from 14 different projects, including newcomers, dropouts, and experienced members. To analyze the data we used procedures of Grounded

978-1-4799-4223-7/14 $31.00 © 2014 IEEE
DOI 10.1109/SBES.2014.9

Theory [14]. From these sources, we obtained three different models that were compiled and organized in one single model. The resulting model comprises 58 barriers organized in six categories.

This paper is structured as follows. Section II introduces the related research; Section III, the research method; Section IV, the results; and Section V, the conclusions and future work.

II. 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 [15], [16] and on OSS projects [17–21]. Dagenais et al. [3] and Begel and Simon [22] present studies regarding newcomers joining process in software projects, but their focus is in industrial settings.

Von Krogh et al. [20] analyzed interviews with developers, emails, source code repository, and documents of the FreeNet project. The authors proposed a joining script for developers who want to take part in the project. Nakakoji et al. [23] studied four OSS projects to analyze the evolution of their 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. 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. Zhou and Mockus [24] 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. [18] 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. [25] 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. [26] 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 [21] 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 [27] 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. [19] and Canfora et al. [17] 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. However, we could not find any study focused on identifying and organizing the barriers faced by newcomers to OSS. In previous work, we report some preliminary results of this research. In [13] we report the results of the systematic literature review, which is part of the current study, and in [28] we report the results of the analysis of the feedback from students and of the answers to an open-question sent to 9 OSS projects. In this paper, we present a qualitative study and a model that categorizes and describes the barriers faced by newcomers when they are onboarding to OSS projects.

III. Research Method

We conducted a qualitative study relying on different data sources to identify and understand the barriers that hinder newcomers' onboarding to OSS projects. One source was a set of studies identified in a systematic literature review on barriers faced by newcomers to OSS projects. The other three sources consisted of data obtained from: feedback from students after they tried to onboard OSS projects; responses to an open-ended question sent to OSS communities; and interviews with newcomers and experienced members of OSS projects.

Three models emerged from the analysis of the data. We compiled these models generating a model of barriers for newcomers to OSS. Figure 1 depicts the method followed. In the following, we detailed the method of the systematic review and of the data collection from the practitioners.

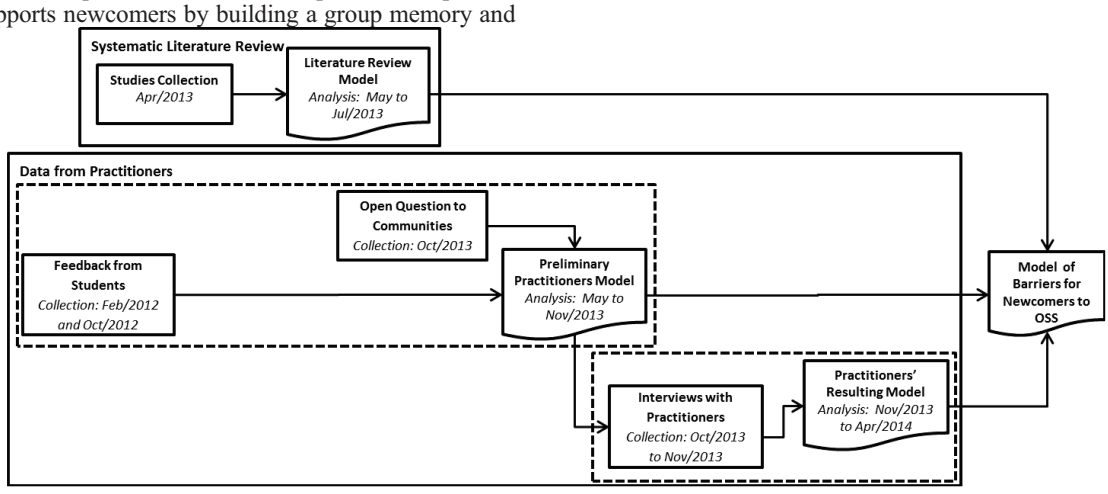

Figure 1: Research Method

A. Systematic Literature Review

We conducted a systematic literature review (SLR) to identify the barriers faced by newcomers empirically evidenced and reported by the literature. The goal of this study was to come up with a list of the barriers encountered by newcomers that can influence their first contributions to the project. Furthermore, we aggregated the problems evidenced by the different studies in a single model.

We undertook a systematic literature Preview (SLR) based upon guidelines established for the Software Engineering domain [29], [30]. In this section, we provide a summary of the protocol used in the SLR. More details about the SLR can be found in [13]. To perform our SLR, we defined the question: *What are the barriers that influence newcomers' onboarding to OSS projects?*

Based on the research question we built a query and retrieved the studies from the ACM, IEEE, Scopus, and Springer Link digital libraries. The search was performed in April 2013.

For each selected paper obtained, we conducted snowball sampling [31] checking if the authors of the selected studies published other relevant studies not retrieved from the digital libraries. We checked their profiles in ACM, IEEE, DBLP, and personal homepages (when available).

The results of the selection and screening are as follows. After running the query on the digital libraries systems, we got 291 candidate papers. For each paper, two independent researchers analyzed title, abstract, and keywords. In a consensus meeting, we came to 33 candidate papers. We checked other papers published by the authors of these 33 candidate studies, finding 20 other candidate papers. After analyzing the abstract of these papers, we selected nine relevant papers, coming to 42 candidate papers. After further analysis, 21 papers were considered relevant and were considered to extract relevant data.

Then, we read the full documents of the primary studies identified and applied open coding to classify the barriers. The analysis resulted in a Literature Review Model of barriers, which are presented in Section IV.A.

B. Data From Practitioners

This section presents the method for the analysis using data from practitioners.

1) Data Collection

We gathered data from three different sources:

- **Source 1:** feedback from students that contributed to OSS projects;
- **Source 2:** answers to an open question sent to developers' mailing lists of OSS projects
- **Source 3:** semi-structured interviews conducted with newcomers and members of OSS projects.

The first source (Source 1) consisted of feedback received from four PhD candidates and five undergrad 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 the last semester of Internet Systems course, therefore, about to join the software development industry. PhD candidates and undergrad students attended to different courses, but received the same assignment: significantly contribute to an OSS project. The contribution should include bug fixes and/or new features implementation.

The students contributed to the JabRef (2 graduate/2 undergraduate), LibreOffice (2 undergraduate), and Mozilla Firefox (3 graduate) projects. After the conclusion of the assignment, their feedback was collected by means of an open-ended questionnaire. We created a questionnaire 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 data was collected in two different moments: the report from graduate students was collected in February 2012; and from undergrad students in October 2012.

The second data source (Source 2) was composed of answers to a questionnaire sent to contributors of OSS projects. The 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 9 different projects: atunes, audacity, LibreOffice, Apache OpenOffice, Mozilla Firefox, jEdit, OpenVPN, FreePlane and emacs. We chose projects from different business domains. It is important to notice that none of them delivers 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 the technology and domain inherent to the project.

The questionnaire delivered to the community members comprised 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 /	6	25.00%
Apache OpenOffice	3	12.50
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%

The final data collection (Source 3) was done by means of semi-structured interviews with practitioners. Semi-structured interviews include a mixture of open-ended and specific questions, designed to elicit not only the information foreseen, but also unexpected types of information [32]. The reason to conduct interviews was to complement the findings gathered from sources 1 and 2, deepening and broadening the understanding about the barriers faced by newcomers.

We recruited subjects that belong to four different groups:

- **Experienced members:** project owners, managers, or developers allowed to commit code directly to the software repository for more than one year.
- **Newcomers that succeeded**: participants that started to contribute to the project less than one year before the interview.
- **Dropout Newcomers**: volunteers that tried to contribute to the project, but gave up;
- **Onboarding Newcomers**: volunteers that were trying to place their first contributions.

The participants were recruited primarily through mailing list and forums from 14 different projects. We also invited the different types of newcomers directly, identifying them by mining and following projects' mailing lists and issue trackers. Only adults 18 years of age and older were eligible to participate in this study. We made no distinction related to gender or nationality. Participants should had software development experience, because we were interested in the barriers to onboard a project and not to learn how to program. Participants are also required to understand and speak English since the interviews were conducted in English.

We interviewed 36 participants from 14 different projects (Pardus, TextMate, zxing, Gephi, Hadoop, jEdit, Moodle, Integrade, Noosfero, OpenOffice, cogroo, etherpad, JabRef, and LibreOffice), including 11 experienced members, 16 newcomers that succeeded, 6 dropout newcomers, and 3 newcomers that were still trying to place their first contributions. Table 3 shows the profile of the interviewees. The interviews were conducted from October 2013 to March 2014.

We used a semi-structured format, in which a script (interview guide) supported the interviewing process. We started with pilot interviews with a five developers involved in Open Source Software Development to adjust the script. After that, we recruited the subjects and conducted the interviews. All the interviews were conducted using textual based chat tools, like Google Talk. We chose this mean because it is an usual mean of communication the participants use in their work, and it facilitates data collection and interviews scheduling.

Each interview was individually conducted and the data was saved in a local computer. Interviews began with a short explanation of the research, followed by some questions to profile the interviewees regarding their technical experience, and main occupation. After that, we conducted the interviews according to the script. The questions served to guide the interview, and were not necessarily asked directly.

Table 3. Profile of the participants

	Time spent per week in OSS	*First Project?*	*Profile*	*Country*	*Years of experience in the project*
P1	less than 5 hours	N	experienced	France	8
P2	from 5 to 10 hours	Y	experienced	Germany	3
P3	from 10 to 20 hours	N	experienced	Germany	3
P4	from 5 to 10 hours	N	experienced	Canada	10
P5	from 5 to 10 hours	N	experienced	Germany	15
P6	more than 20 hours	N	experienced	Hungary	10
P7	more than 20 hours	N	experienced	Australia	5
P8	more than 20 hours	N	experienced	Brazil	5
P9	more than 20 hours	N	experienced	Turkey	8
P10	from 5 to 10 hours	N	experienced	Brazil	15
P11	less than 5 hours	N	experienced	Brazil	7
P12	less than 5 hours	Y	newcomer	Germany	0
P13	less than 5 hours	Y	newcomer	Brazil	0
P14	from 5 to 10 hours	Y	newcomer	India	1
P15	from 5 to 10 hours	Y	newcomer	India	0
P16	less than 5 hours	Y	newcomer	Germany	0
P17	less than 5 hours	N	newcomer	USA	0
P18	less than 5 hours	Y	newcomer	USA	0
P19	more than 20 hours	Y	newcomer	Greece	0
P20	less than 5 hours	Y	newcomer	Brazil	0
P21	less than 5 hours	Y	newcomer	Brazil	0
P22	less than 5 hours	Y	newcomer	Brazil	0
P23	N/I	N	newcomer	UK	0
P24	from 10 to 20 hours	N	newcomer	Brazil	1
P25	from 5 to 10 hours	Y	newcomer	Brazil	1
P26	N/I	Y	newcomer	France	0
P27	from 5 to 10 hours	N	newcomer	Germany	0
P28	from 5 to 10 hours	N	dropout	USA	0
P29	less than 5 hours	Y	dropout	India	0
P30	less than 5 hours	N	dropout	Germany	0
P31	less than 5 hours	Y	dropout	Brazil	0
P32	less than 5 hours	Y	dropout	India	0
P33	less than 5 hours	Y	dropout	India	0
P34	less than 5 hours	N	onboarding	China	0
P35	from 10 to 20 hours	Y	onboarding	India	0
P36	less than 5 hours	Y	onboarding	Greece	0

2) Data Analysis

We qualitatively analyzed the data using procedures of Grounded Theory (GT) [14]. According to Seaman [32], a grounded approach enables the identification of new concepts, making it a valid choice for software engineering research. GT is based in the concept of coding. Coding means attaching codes, or labels, to pieces of text which are relevant to a particular theme or idea, grouping and examining the ideas to explain a phenomena [32]. Coding can be divided into three steps: open coding, where concepts are identified and their properties and dimensions are discovered; axial coding, where connections among 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. We applied just the open and axial coding, because our goal was to identify barriers. The coding was performed using the ATLAS.ti[1] tool.

Although the purpose of the GT method is the construction of substantive theories, its use does not necessarily need to remain restricted only to researches with this goal. According to Strauss and Corbin [14], a researcher may use only some of its procedures to meet one's research goals.

[1] http://www.atlasti.com

We split our analysis in two steps. The first (preliminary) step (QS1) consisted of the analysis of data from Sources 1 and 2, open coding and axial coding this data. In the second step (QS2), the codes and categories found in QS1 were used as seeds for the coding of data from Source 3. During open coding, we assigned codes to sentences, paragraphs, or revisions. This procedure overlapped the axial coding, in which we identified connections between the categories. We executed open and axial coding several times to refine the emerging codes and categories.

For the first step, the open coding process was conducted in parallel by three researchers. Each researcher quoted and coded the documents independently. After coding, the researchers discussed the quotes and codes until they came to a consensus for the whole set of documents. This was done to mitigate the bias eventually caused by the participation of a single researcher in the coding process. After coding, we discussed the quotes and codes until coming 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. The result of this step was a Preliminary Qualitative Model of the barriers faced by newcomers to OSS.

For the second step, we analyzed the data obtained from the interviews. The process of analysis was similar to the one applied in the first step. However, we used the codes and categories identified in the first step as seeds to the open coding. Moreover, only one researcher conducted open and axial coding. Some iteration with other two other researchers were conducted in order to discuss and review the codes and categories. As the work progressed, new categories and codes appeared and some codes were merged, because the researchers' comprehension evolved and new information sprang up.

We provide more details on the collection and analysis of data from practitioners in a technical report available at http://www.igor.pro.br/publica/TR/SBES2014_TR.pdf.

IV. Results and Discussion

In this section, we report each model separately, and then, the resulting combined model.

A. SLR Model

During the SLR, we analyzed 21 studies. From these studies, we identified 16 barriers, grouped in five categories: *Social Interactions, Newcomers' Previous Knowledge, Finding a Way to Start, Documentation Problems, and Code Issues*. Table 4 shows the barriers identified for each category and the studies that evidenced them. The categories are briefly described in the following.

Social interaction issues. This category grouped the barriers related to the way newcomers interact with the community, including issues related to who were the members they exchange messages with, the size of their contact network, how they communicate, and how the community communicate with them. These barriers were mostly evidenced from historical data mined from software repositories.

Newcomers' Previous Knowledge. This category comprised problems related to the experience of the newcomers regarding the project. It includes domain, process, and technical previous skills.

Table 4. Studies that evidence each barrier

Category	Barrier	Studies
Social Interaction Issues	Socialization of newcomers and project members	[1], [24], [33–37]
	Newcomers do not receive (timely and proper) response	[18], [20], [24], [25], [38–40]
	Newcomers do not send a correct/meaningful message	[24], [38]
	Finding Help - Mentor/Expert	[17], [26], [39]
Newcomers' Previous Knowledge	Lack of domain expertise	[20], [40]
	Lack of previous technical experience	[20], [24], [33–35], [38], [41]
	Lack of knowledge on project processes and practices	[41]
Finding a Way to Start	Finding an appropriate task/issue to start with	[20], [27], [42], [43]
	Finding the correct artifacts to fix an issue	[26]
Documentation Problems	Outdated documentation	[39], [40]
	Code comments not clear	[40]
	Information overload	[26], [27], [40]
	Lack of documentation/diagrams	[40]
Code Issues	Code complexity/instability	[34], [44]
	Problems to understand architecture/code structure	[26], [27], [40]
	Issues setting up a local workspace	[40]

Finding a way to start. Newcomers need support to find a task and the proper artifacts to change. We found that, from communities perspective, newcomers should be able to find the most appropriate task themselves [20]. However, some studies showed that the newcomers need special attention [42], [43].

Documentation Problems. Refers to needs to learn technical and social aspects of the project to be able to contribute. A rich and up-to-date documentation is essential for newcomers trying to understand a project. However, just providing a bunch of documentation leads to information overload. Finding outdated documentation or getting lost in a huge amount of information can lead to demotivation.

Code Issues. Comprised the barriers related to the source code of the products. To contribute, newcomers need to change or interact with existing source code. Therefore, it is necessary for the newcomers to have enough knowledge about the code to start their contributions. The main complaint regarding code was that its structure was hard to understand, and learning it takes too much time. A study evidenced that newcomers had difficulties to set up their environment [40].

The category more thoroughly studied is *Social Interaction Issues*, accounting for 15 studies, followed by *Newcomers' Previous Knowledge*, with eight studies. The other categories ranged from four to six related studies each. It was possible to notice that the literature focused on the social issues of newcomers' onboarding. The technical barriers, like understanding code/architecture, dealing with versioning system, setting up workspace, building, and standards were poorly or not studied so far.

Due to the nature of the approach to establish the model, there was at least one paper associated to any problem. Considering the most studied one, we found that the most evidenced problems are newcomers' previous technical experience and aspects regarding social network characteristics and response reception.

B. Preliminary Practitioners' Model

As presented in Figure 1, the Preliminary Qualitative Model was the result of the analysis conducted from Sources 1 and 2.

The Preliminary Qualitative Model comprised seven categories along with 33 barriers. Table 5 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 of answers to the open question in which that category appeared. In the following, we briefly present each category, including tables that report from which source we observed the barriers. Moreover, for Source 2, we split the evidence according to the respondent experience.

Table 5. Overview of Categories that Emerged

Category	# of documents (feedback/ question)	# quotes (feedback/ question)	# of barriers
Social interaction issues	11 (6 / 5)	12 (8/4)	4
Newcomers' behavior	3 (0 / 3)	3 (0/3)	2
Newcomers' technical knowledge	12 (4 / 8)	16 (7/9)	5
Finding a way to start	11 (8 / 3)	22 (18/4)	3
Documentation problems	15 (8 / 7)	23 (15/8)	10
Code issues	15 (7 / 8)	21 (11/10)	5
Issues setting up workspace	8 (4 / 4)	15 (10/5)	4

Social Interactions Issues. In Table 6 we can observe that social issues are reported by the students and by community members that joined the projects recently. Two barriers evidenced in the SLR did not appear in these studies. On the other hand, the other barriers confirmed the evidence found in the SLR. Moreover, three barriers evidenced were related to the way newcomers are received by the community, detailing a barrier found in the SLR – *Receive (timely and proper) answer*.

Table 6. Social Interaction barriers quotes per data source and time in the project

Data Source / Background/ Time in the project	Source 1: Feedback Students	Source 2: Open Questions		
		Less than 6 months	Between 6 months and 3 years	More than 3 years
Delayed answers	•			
Impolite answers	•			
Finding someone to help	•	•		
Community uses intimidating terms		•		

Newcomers' Behavior. From the answers to the open question, we identified issues related to newcomers' behavior that can hinder their onboarding. We identified two barriers under this category, as presented in Table 7.

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

Data Source / Background/ Time in the project	Source 1: Feedback Students	Source 2: Open Questions		
		Less than 6 months	Between 6 months and 3 years	More than 3 years
Lack of Commitment		•		•
Underestimating the challenge				•

Newcomers' Technical Knowledge. Reported as a barrier in the SLR, *newcomers' technical knowledge* appeared as a category in this model. Five barriers were identified in 12 documents analyzed, detailing this category. Both newcomers and community members recognized previous knowledge as a barrier that hindered newcomers' onboarding, as it can be observed in Table 8.

Table 8. Newcomers' Technical Knowledge barriers per data source and time in the project

Data Source / Background/ Time in the project	Source 1: Feedback Students	Source 2: Open Questions		
		Less than 6 months	Between 6 months and 3 years	More than 3 years
Lack of previous knowledge on project tooling	•		•	
Lack of knowledge on versioning control system		•		•
Choosing the right development tools		•		
Lack of knowledge on technologies used	•			
Lack of knowledge on the programming language used	•			

Finding a way to start with. In this category, the barriers found are the same as those identified in the SLR. The only exception was the problem related to *outdated list of bugs*, which was reported during students' feedback sessions. In Table 9, we present the evidenced barriers according to the data sources and profiles of the respondents. We can see that students that were onboarding to 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	Source 1: Feedback Students	Source 2: Open Questions		
		Less than 6 months	Between 6 months and 3 years	More than 3 years
Finding the right piece of code to work with	•	•		
Outdated list of bugs	•			
Finding a task to start with	•	•		•

Documentation problems. Problems related to documentation were recurrently reported. *Unclear documentation* and *spread documentation* were mentioned as barriers. *Lack of documentation* was specialized, resulting in seven barriers. In total, we identified ten barriers under this category. Table 10 reports the barriers and who reported them.

Table 10. Problems with documentation per data source and time in the project

Data Source / Background/ Time in the project	Source 1: Feedback Students	Source 2: Open Questions		
		Less than 6 months	Between 6 months and 3 years	More than 3 years
Outdated documentation	•		•	
Unclear documentation	•			
Spread documentation			•	
Lack of documentation	•	•	•	
Lack of documentation on project structure	•			
Lack of documentation on setting up workspace	•			
Lack of documentation on Contribution Process	•			•
Lack of code comments		•		
Lack of design documentation		•		
Lack of code documentation			•	

Code issues. Problems related to code were also identified in the feedback from students and open questions. However, in these studies, we identified different barriers from those found in the SLR. The only exception is the cognitive barrier, related

to *problems to understand the architecture/code structure*. Table 11 presents the barriers split according to the data source.

Table 11. Code issues reported per data source and time in the project

Data Source / Background/ Time in the project	Source 1: Feedback Students	Source 2: Open Questions		
		Less than 6 months	Between 6 months and 3 years	More than 3 years
Bad code quality	•		•	•
Codebase size		•	•	•
Outdated code		•		
Problems understanding the code	•	•	•	
Lack of code standards	•			

Issues setting up the workspace. To modify the application it is necessary to build the application locally first, what can take time and demotivate the newcomer. Differently from the SLR, *issues setting up the workspace* appeared as a category, encompassing four barriers. This category appeared in eight documents, and was related to the barriers presented in Table 12. In the table, it is also possible to observe the data source from which the barriers were evidenced.

Table 12. Setting up workspace barriers per data source and time in the project

Data Source / Background/ Time in the project	Source 1: Feedback Students	Source 2: Open Questions		
		Less than 6 months	Between 6 months and 3 years	More than 3 years
Issues setting up a local workspace	•	•	•	
Platform dependency	•		•	
Difficulty to find the correct source code	•			
Library dependencies	•			

C. Practitioners' Resulting Model

The result of the analysis was the emergence of 50 barriers grouped in 6 categories. Some of them also presented subcategories. The categories are briefly described in the following. As in the previous section, for each category, we present tables showing the profile of the interviewees who provided the evidence for the barriers.

Reception Issues. The receptivity of OSS communities was also evidenced as a barrier, which even lead newcomers to give up. This category comprises the barriers related to the interactions that occur between newcomers and the community. A breakdown during these social interactions can lead to demotivation and result in newcomers' dropping out. We could identify four barriers, presented in Table 13. The barriers can be compared to the barriers identified in the social interaction categories in the previous studies.

Table 13. Barriers that emerged from a qualitative analysis of data categorized as "reception issues"

Barriers	Dropout	Newcomers	Experienced
Newcomer receive an answer that was not "newcomer friendly"		•	•
Delay to receive a response		•	•
Not receiving an answer			•
Impolite messages		•	•

Newcomers' characteristics. This category comprises two other subcategories: *newcomers' behavior*, with ten barriers (Table 14); *and newcomers' technical background* (Table 15), with five barriers. During the interviews, we identified many barriers related to *newcomers' behavior* that were not found in the previous studies. Experienced members reported the most part of them. Regarding *newcomers' technical background*, the interview analysis confirmed the barriers identified in other studies.

Table 14. Barriers that emerged from a qualitative analysis of data categorized as "newcomers' behavior"

Barriers	Dropout	Newcomers	Experienced
Lack of proactivity	•	•	•
Need to be patient		•	
Underestimate the challenge			•
Lack of commitment			•
Not acknowledging/thanking answers			•
Shyness			•
English level			•
Making useless comments in the mailing list/forums			•
Low responsiveness			•
Not sending a correct meaningful and correct message			•

Table 15. Barriers that emerged from a qualitative analysis of data categorized as "newcomers' technical background"

Barriers	Dropout	Newcomers	Experienced
Lack of proper knowledge in the programming language		•	•
Lack of knowledge on technologies and tools used by the project		•	•
Lack of previous knowledge on versioning control system		•	•
Lack of experience on unit testing		•	
Difficulty choosing the right development tools		•	

Newcomers Need Orientation. We found that newcomers often face unfamiliar and rugged landscapes when onboarding to OSS project. They need proper orientation to find their way in the project, and correctly place their contributions. We identified at least one barrier belonging to this category in 20 interviews. The barrier *difficulty to find a mentor* was mentioned previously (*difficulty to find someone to help*) under *social interaction issues* category.

Table 16. Barriers that emerged from a qualitative analysis of data categorized as "find a way to start"

Barriers	Dropout	Newcomers	Experienced
Finding a task to start with	•	•	•
Reproducing issues	•		
Finding the right piece of code to work	•	•	•
Finding a mentor	•	•	•
Poor "How to contribute"	•	•	•
Newcomers don't know what is the contribution flow		•	

Documentation problems. Regarding documentation problems, we found ten barriers, as presented in Table 17. The barriers identified in the interviews had been already evidenced in other data sources.

Table 17. Barriers that emerged from a qualitative analysis of data categorized as "documentation problems"

Barriers	Dropout	Newcomers	Experienced
Spread documentation		•	
Outdated Documentation	•	•	•
Code comments not clear		•	
Lack of documentation in general			•
Lack of code comments		•	
Lack of code documentation	•	•	•
Lack of design documentation / code structure		•	
Lack of documentation on setting up workspace		•	

Technical Hurdles. This category presented the highest number of barriers evidenced during the analysis. We put all the problems faced by newcomers while dealing with the source code in a single category. To do so, we further classified these barriers into three subcategories: *Code/architectural hurdles* (Table 18), with seven barriers; *Hurdles to submit changes* (Table 19), with four barriers; and *Local environment setup hurdles* (Table 20), with four barriers.

Table 18. Barriers that emerged from a qualitative analysis of data categorized as "code/architecture hurdles"

Barriers	Dropout	Newcomers	Experienced
Bad design quality		•	•
Bad code quality			•
Code complexity		•	•
Codebase Size	•	•	•
Understanding the architecture/code structure		•	•
Understanding the code	•	•	•
Understanding flow of information		•	

Table 19. Barriers that emerged from a qualitative analysis of data categorized as "hurdles to submit changes"

Barriers	Dropout	Newcomers	Experienced
Delay to get contribution accepted/reviewed			•
Getting contribution accepted	•		•
Lack of information on how to send a contribution		•	•
Issue to create a patch			•

Table 20. Barriers that emerged from a qualitative analysis of data categorized as "local environment setup hurdles"

Barriers	Dropout	Newcomers	Experienced
Building workspace locally	•	•	•
Library dependencies	•		
Platform dependence		•	
Finding the correct source code		•	

Cultural Differences. Once OSS development is a case of global software development, people from different cultures need to collaborate. These differences can result in interaction problems. In our analysis, three subjects reported that some newcomers face cultural barriers while onboarding. We could find two barriers under this category, shown in Table 21.

Table 21. Barriers that emerged from a qualitative analysis of data categorized as "cultural differences"

Barriers	Dropout	Newcomer	Experienced
Some newcomers need to contact a real person			•
Message received is considered rude		•	

D. Resulting model of barriers for newcomers to OSS projects

After obtaining the model from the analysis of interviews, we iteratively reanalyzed the models obtained from all sources, relying on their respective data. The goal of this reanalysis was to combine the findings to create a single model accommodating all the barriers evidenced. Once again, we merged some barriers and reorganized the categories.

The resulting model aggregates all the barriers evidenced in the intermediate models. The model was obtained after the composition of the analysis of axial coding. Each leaf code is a concept grounded in the data found during open coding.

The model presents 58 barriers, organized in 6 categories and in several subcategories. The model is presented in Figure 2. The numbers after the name of each barrier correspond to the amount of times (different documents) that the barrier had been identified per source. The numbers represent in this order: number of studies from the SLR that evidenced the barrier (out of 21); number of students that reported the barrier in their feedback (out of 9); number of mentions in the open questions (out of 24); and number of interviewees that reported the barrier (out of 36). In parenthesis, we also provided the number of projects in which the barriers were evidenced, considering only the data from practitioners.

As it is possible to notice, we highlight the barriers which evidence appeared in all four data sources, including reports from practitioners recruited in different ways and evidence from the current literature.

V. Threats To Validity

Although we analyzed data from a variety of sources, and from different projects, it is very likely that we did not reach all possible barriers and explanation of the barriers. We are aware that each project has its singularities, so, the level of support and the barriers can differ according to the project. Our strategy to consider different projects and different profiles of developers aimed at alleviating this issue, identifying recurrent mentions of barriers from multiple perspectives.

Another threat to the validity of the results is the subjectivity of the data classification. We used the Grounded Theory procedures to mitigate this threat, given that the GT requires the entire analysis to be grounded in the data collected. Additionally, the analysis process was discussed along with two other researchers, to encourage a better validation of the interpretations through the mutual agreement.

As we sent open invitations to mailing list, there should be sampling bias in our interviewees and open question respondents, namely self-selection bias and social desirability bias. But, getting different sources and analyzing the answers in context to identify specificities, we tried to avoid that effect.

VI. Conclusions and Future Work

In this paper, we reported the results of a qualitative study relying on data obtained from a systematic literature review and from practitioners. The main contribution is a model of barriers that hinder newcomer onboarding to OSS projects, composed of 58 barriers grouped in six different categories. This model organizes the existing common knowledge about barriers faced by newcomers to OSS projects.

A fact to notice is that 50 out of 58 barriers presented in the resulting model were identified in the data from interviews with practitioners. We also could notice that less than 30% of the barriers (17 barriers after reanalysis) were evidenced by the literature. Moreover, only six barriers presented in the model were evidenced in all the sources analyzed.

The OSS communities can benefit from these results to provide appropriate support to newcomers. We expect to make communities aware of the problems that can hinder the first contributions, offering them an opportunity to think about the reception of newcomers.

During our interviews, we found that experienced members are interested in the results of our research to offer a better support to newcomers. Existing members of the projects can think about maps or signs to orient newcomers and guide them, or, at least, warn them about the barriers they can find. We

believe that simple actions can make a great impact. By making newcomers aware of the problems they can face and of the strategies used by the project to support each of the categories (or barriers), the communities can manage newcomers' expectations and projects can benefit from more contributions.

This research topic needs further exploration and can bring fruitful results not only for OSS projects, but also for software projects in general. We believe that our results can offer insights for researching ways to facilitate the influx of newcomers to OSS projects. Each of the categories presented can foster further research and enable investigations in different perspectives.

ACKNOWLEDGEMENTS

The authors would like to thank Fundação Araucária, CNPq (477831/2013-3), FAPEAM, NAPSoL-PRP-USP, NAWEB, CAPES (BEX 2038-13-7) and FAPESP for financial support.

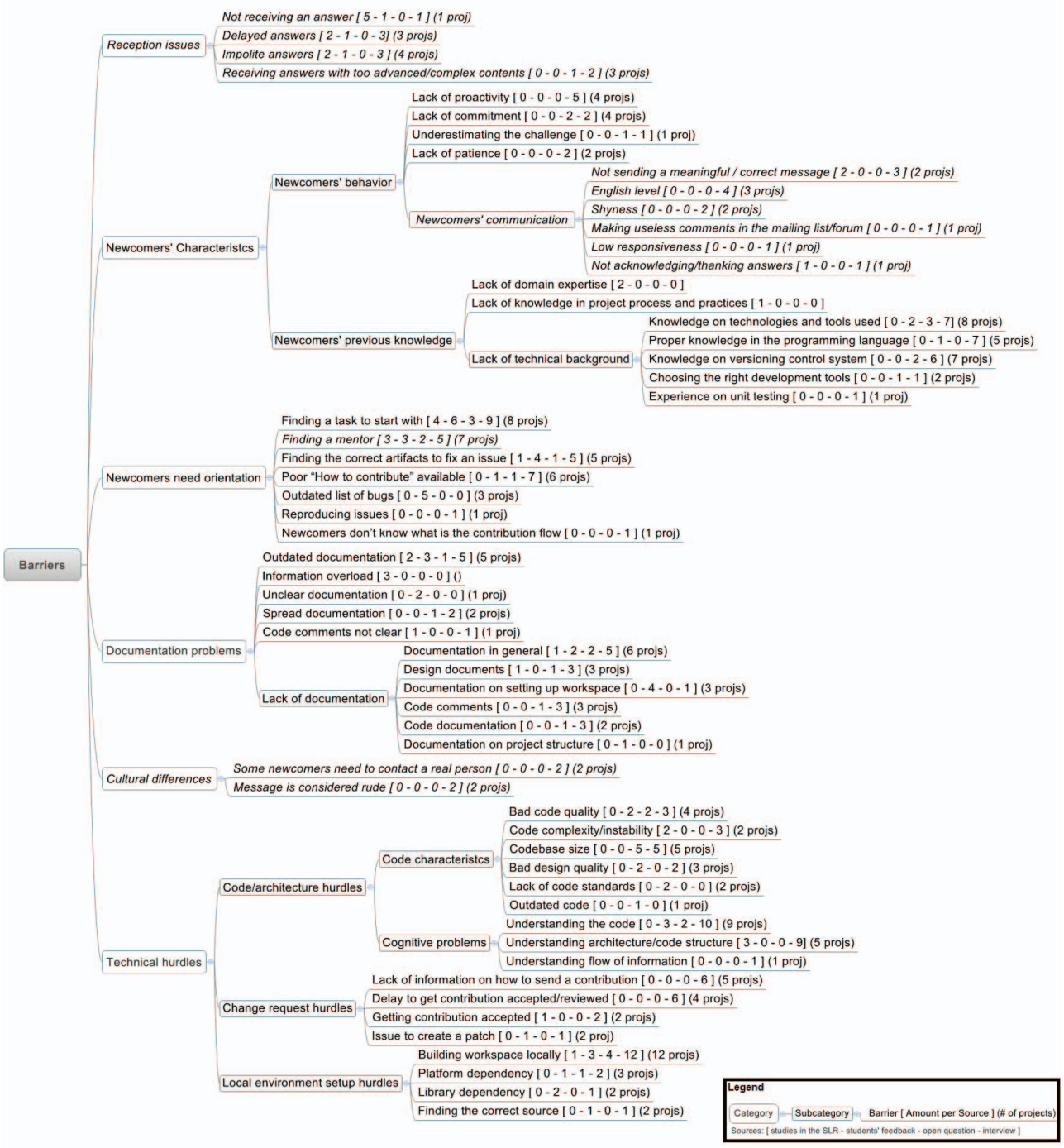

Figure 2. Model of barriers for newcomers to OSS

REFERENCES

[1] I. Qureshi and Y. Fang, "Socialization in Open Source Software Projects: A Growth Mixture Modeling Approach," *Org. Res. Methods*, vol. 14, no. 1, pp. 208–238, 2011.

[2] R. E. Kraut, M. Burke, J. Riedl, and P. Resnick, "The Challenges of Dealing with Newcomers," MIT Press, 2012, pp. 179–230.

[3] B. Dagenais, H. Ossher, R. K. E. Bellamy, M. P. Robillard, and J. P. de Vries, "Moving into a new software project landscape," in *32nd International Conference on Software Engineering*, 2010, pp. 275–284.

[4] W. Scacchi, "Understanding the requirements for developing open source software systems," *IEE Proceedings Software*, vl. 149, no. 1, pp. 24-39, 2002.

[5] I. Herraiz, G. Robles, J. J. Amor, T. Romera, J. M. G. Barahona, and J. Carlos, "The processes of joining in global distributed software projects.," in *2006 International Workshop on Global Software Development for the Practitioners*, 2006, pp. 27–33.

[6] I. Steinmacher, M. A. Gerosa, and D. Redmiles, "Attracting, Onboarding, and Retaining Newcomer Developers in Open Source Software Projects," in *Workshop on GSD in a CSCW Perspective*, 2014.

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

[8] C. Santos, G. Kuk, F. Kon, and J. Pearson, "The Attraction of Contributors in Free and Open Source Software Projects," *J. Strateg. Inf. Syst.*, vol. 22, no. 1, pp. 26–45, Mar. 2013.

[9] S. K. Shah, "Motivation, Governance, and the Viability of Hybrid Forms in Open Source Software Development," *Manage. Sci.*, vol. 52, no. 7, pp. 1000–1014, 2006.

[10] Y. Ye and K. Kishida, "Toward an Understanding of the Motivation Open Source Software Developers," in *25th International Conference on Software Engineering*, 2003, pp. 419–429.

[11] V. Wolff-Marting, C. Hannebauer, and V. Gruhn, "Patterns for tearing down contribution barriers to FLOSS projects," in *12th Intl. Conf. on Intelligent Software Methodologies, Tools & Techniques*, 2013, pp. 9-14.

[12] K. Fogel, *Producing Open Source Software: How to Run a Successful Free Software Project*, First. O'Reilly Media, 2013.

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

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

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

[16] P. Vora, N. Komura, and S. U. Team, "The n00b Wikipedia Editing Experience," in *6th Intl. Symposium on Wikis and Open Collaboration*, 2010, pp. 36:1–36:3.

[17] G. Canfora, M. Di Penta, R. Oliveto, and S. Panichella, "Who is Going to Mentor Newcomers in Open Source Projects?," in *20th International Symposium on the Foundations of Software Engineering*, 2012, pp. 1-11.

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

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

[20] G. Von Krogh, S. Spaeth, and K. R. Lakhani, "Community, joining, and specialization in open source software innovation: A case study," *Research Policy*, vol. 32, no. 7, pp. 1217–1241, 2003.

[21] J. Wang and A. Sarma, "Which bug should I fix: helping new developers onboard a new project," in *4th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)*, 2011, pp. 76–79.

[22] A. Begel and B. Simon, "Novice Software Developers, All over Again," in *4th Intl. Workshop on Computing Education Research*, 2008, pp. 3-14.

[23] K. Nakakoji, Y. Yamamoto, Y. Nishinaka, K. Kishida, and Y. Ye, "Evolution Patterns of Open-source Software Systems and Communities," in *Workshop on Principles of Software Evolution*, 2002, pp. 76–85.

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

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

[26] D. Cubranic, G. C. Murphy, J. Singer, and K. S. Booth, "Hipikat: a project memory for software development," *IEEE Transactions on Software Engineering*, vol. 31, no. 6, pp. 446–465, 2005.

[27] Y. Park and C. Jensen, "Beyond pretty pictures: Examining the benefits of code visualization for open source newcomers," in *5th IEEE International Workshop on Visualizing Software for Understanding and Analysis*, 2009, pp. 3–10.

[28] I. Steinmacher, I. S. Wiese, T. Conte, M. A. Gerosa, and D. Redmiles, "The Hard Life of Open Source Software Project Newcomers," in *International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)*, 2014.

[29] B. Kitchenham and P. Brereton, "A systematic review of systematic review process research in software engineering," *Information and Software Technology*, vol. 55, no. 12, pp. 2049–2075, Dec. 2013.

[30] B. Kitchenham and S. Charters, "Guidelines for performing Systematic Literature Reviews in Software Engineering," Keele University and Durham University, EBSE 2007-001, 2007.

[31] S. Jalali and C. Wohlin, "Systematic Literature Studies: Database Searches vs. Backward Snowballing," in *International Symposium on Empirical Software Engineering and Measurement*, 2012, pp. 29–38.

[32] C. B. Seaman, "Qualitative methods in empirical studies of software engineering," *Software Engineering, IEEE Transactions on*, vol. 25, no. 4, pp. 557–572, Jul. 1999.

[33] C. Bird, "Sociotechnical coordination and collaboration in open source software," in *27th IEEE International Conference on Software Maintenance*, 2011, pp. 568–573.

[34] C. Bird, A. Gourley, P. Devanbu, A. Swaminathan, and G. Hsu, "Open Borders? Immigration in Open Source Projects," in *ICSE Workshops MSR '07. Fourth International Workshop on Mining Software Repositories, 2007*, 2007, pp. 6–6.

[35] N. Ducheneaut, "Socialization in an Open Source Software Community: A Socio-Technical Analysis," *Computer Supported Cooperative Work (CSCW)*, vol. 14, no. 4, pp. 323–368, 2005.

[36] P. He, B. Li, and Y. Huang, "Applying Centrality Measures to the Behavior Analysis of Developers in Open Source Software Community," in *Cloud and Green Computing (CGC), 2012 Second International Conference on*, 2012, pp. 418–423.

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

[38] V. Singh, "Newcomer integration and learning in technical support communities for open source software," in *17th ACM international conference on Supporting group work*, 2012, pp. 65–74.

[39] I. Steinmacher, I. S. Wiese, and M. A. Gerosa, "Recommending mentors to software project newcomers," in *Third International Workshop on Recommendation Systems for Software Engineering*, 2012, pp. 63–67.

[40] K.-J. Stol, P. Avgeriou, and M. Ali Babar, "Identifying architectural patterns used in open source software: approaches and challenges," in *14th international conference on Evaluation and Assessment in Software Engineering*, 2010, pp. 91–100.

[41] A. Schilling, S. Laumer, and T. Weitzel, "Who Will Remain? An Evaluation of Actual Person-Job and Person-Team Fit to Predict Developer Retention in FLOSS Projects," in *2012 45th Hawaii International Conference on System Sciences*, 2012, pp. 3446–3455.

[42] X. Ben, S. Beijun, and Y. Weicheng, "Mining Developer Contribution in Open Source Software Using Visualization Techniques," in *2013 Third International Conference on Intelligent System Design and Engineering Applications (ISDEA)*, 2013, pp. 934–937.

[43] A. Capiluppi and M. Michlmayr, "From the Cathedral to the Bazaar: An Empirical Study of the Lifecycle of Volunteer Community Projects," in *1st International Conference on Open Source Systems*, 2007, pp. 31–44.

[44] V. Midha, P. Palvia, R. Singh, and N. Kshetri, "Improving open source software maintenance," *Journal of Computer Information Systems*, vol. 50, no. 3, pp. 81–90, 2010.