# Supporting newcomers to overcome the barriers to contribute to open source software projects

Igor Fabio Steinmacher

Text submitted
to the
Institute of Mathematics and Statistics
of the
University of São Paulo
for the
Doctoral degree in Science

Program: Computer Science

Advisor: Professor Marco Aurélio Gerosa

This research is supported by Fundação Araucária (projeto DINTER UTFPR/IME-USP), CAPES (proc. BEX 2037-13-7), NAWEB, NAPSoL-PRP-USP, FAPESP and CNPq (Universal 477831/2013-3)

São Paulo, November, 2015

**Supporting newcomers to overcome the barriers to contribute to open source software projects**

This version of the thesis contains the changes suggested by the Committee Members during the public defense of the original version of this work, which occurred in Feb 26th, 2015. A copy of the original version is available at Institute of Mathematics and Statistics (IME) -University of São Paulo (USP)

Committee Members:

- Prof. Dr. Marco Aurélio Gerosa (advisor) - IME-USP
- Prof. Dr. Christoph Treude - UFRN
- Prof. Dr. Tayana Uchoa Conte – UFAM
- Prof. Dr. José Carlos Maldonado – ICMC-USP
- Prof. Dr. Daniel Batista Macêdo – IME-USP

# Acknowledgements

First, I would like to thank my advisor Marco Aurélio Gerosa for the guidance and helpful discussions during all these years, putting me on track whenever I got lost. His way of thinking and supporting me on my research taught me everything I know about being a researcher and conducting research. Marco helped me, guided me, and became a good friend.

I also would like to thank David F. Redmiles, who received and welcomed me at UCI. I had a really good time there and could learn a lot with his suggestions and feedbacks. I am grateful to Tayana Conte and Cleidson de Souza who introduced me the qualitative research and supported me during my journey. I am very grateful to all the members of my qualifying committee and final examining committee. It was a pleasure to receive feedback and comments from a group of outstanding researchers: Carlos Denner, Cleidson de Souza, Fabio Kon, Daniel Batista, Christoph Treude, José Carlos Maldonado and Tayana Conte. I extend this gratitude to all the anonymous reviewers of the scientific community that gave important feedback to preliminary versions of this work reported in scientific papers.

Thank you to all the open source software practitioners who volunteered and our students who participated of our studies. The kindness of complete strangers was essential for my research. I hope that my results bring benefits to OSS communities and to newcomers who want to contribute.

This PhD would not be possible without the interinstitutional doctoral project (DINTER), financed by Fundação Araucária. I would like to thank Fabio Kon and Carlos Eduardo Ferreira for heading it at IME-USP and Reginaldo Ré for helping us coordinating the project at UTFPR. I extend this greeting to all the IME professors who supported and participated in this project. Due to the remote settings of DINTER project, I had to solve many issues while away from São Paulo, and I would not do that without the help of Lucileide R. G. Tadei, Secretary of CCP/CC.

Thanks for all the colleagues from LAPESSC and CCSL, mainly Gustavo Ansaldi, Maurício Aniche and Igor Wiese, for the support, help, and hints during this period. I also would like to thank the other colleagues who were part of this doctorate program, residents of apartment 141: André Kawamoto, André Schwerz, Frank Helbert, Igor Wiese, Ivanilton Polato, Lucio Valentin, Luiz Arthur Feitosa, Marcos Silvano, Rafael Liberato, Rodrigo Campiolo, Rogério Gonçalves e Wellington Previero. I could not forget all the colleagues from Department of Computing at UTFPR who took part of my duties during this period.

During this research, I received financial support from different foundations. I am really grateful to Fundação Araucária (DINTER UTFPR/IME-USP), CAPES Sandwich scholarship (BEX 2037-13-7), NAWEB, NAPSoL-PRP-USP, FAPESP, and CNPq (Universal 477831/2013-3).

Finally, I would like to thank to my Family: my wife, my angel, my supporter Ana Paula Chaves Steinmacher, who encouraged, understood, pushed, and loved me during this long journey; and my two little pumpkins, who arrived during this period, Dante and Alice, who could not understand the reasons of my absence. I love you all. This thesis is for and because of you.

# Abstract

STEINMACHER, I. F. **Supporting newcomers to overcome the barriers to contribute to open source software projects**. 2015. 183 f. Tese (Doutorado) – Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2015

Community-based Open Source Software (OSS) projects are generally self-organized and dynamic, receiving contributions from volunteers spread across the globe. These communities' survival, long-term success, and continuity demand a constant influx of newcomers. However, newcomers face many barriers when making their first contribution to an OSS project, leading in many cases to dropouts. Therefore, a major challenge for OSS projects is to provide ways to support newcomers during their first contribution. In this thesis, our goal was to identify and understand the barriers newcomers face and provide appropriate strategies to lower these barriers. Toward this end, we conducted multiple studies, using multiple research methods. To identify the barriers, we used data collected from: semi-structured interviews with 35 developers from 13 different projects; 24 answers to an open questionnaire conducted with OSS developers; feedback from 9 graduate and undergraduate students after they tried to join OSS projects; and 20 primary studies gathered via a systematic literature review. The data was analyzed using Grounded Theory procedures: namely, open and axial coding. Subsequently, the analysis resulted in a preliminary conceptual model composed of 58 barriers grouped into six categories: cultural differences, newcomers' characteristics, reception issues, newcomers' orientation, technical hurdles, and documentation problems. Based on the conceptual model, we developed FLOSScoach, a portal to support newcomers making their first OSS project contribution. To assess the portal, we conducted a study with undergraduate students, relying on qualitative data from diaries, self-efficacy questionnaires, and the Technology Acceptance Model. By applying the model to a practical application and assessing it, we could evaluate and improve the barriers model, changing it according to improvements identified during the conception of the tool, as well as suggestions received from the study participants. The FLOSScoach study results indicate that the portal played an important role guiding newcomers and lowering barriers related to the orientation and contribution process, whereas it was inefficient in lowering technical barriers. We also found that the portal is useful, easy to use, and increased newcomers' confidence to contribute. The main contributions of this thesis are: (i) empirical identification and modeling of barriers faced by OSS project newcomers; and (ii) a portal providing information to support OSS project newcomers.

**Keywords:** Newcomers; Beginners; Newbies; Novices; Joiner; Open Source Software; Free Software; FLOSS; Barriers; Obstacles; Problems; Joining Process; Onboarding; Socialization; Empirical Software Engineering.

# Table of Contents

# List of Abbreviations

| | |
|---|---|
| AA | Alcoholics Anonymous |
| ACM | Association for Computing Machinery |
| ASF | Apache Software Foundation |
| CCSL | Centro de Competência em Software Livre |
| CHASE | International Workshop on Cooperative and Human Aspects of Software Engineering |
| DBLP | The DBLP Computer Science Bibliography |
| DSD | Distributed Software Development |
| GPL | General Public License |
| GSoC | Google Summer of Code |
| GT | Grounded Theory |
| IEEE | Institute of Electrical and Electronics Engineers |
| IRC | Internet Relay Chat |
| IST | Information and Software Technologies Journal |
| LAPESSC | Laboratório de Pesquisa em Engenharia de Software e Sistemas Colaborativos |
| LoC | Lines of Code |
| LPP | Legitimate Peripheral Participation |
| OSS | Open Source Software |
| PhD | Doctor of Philosophy |
| Q&A | Questions and Answers |
| SBSC | Simpósio Brasileiro de Sistemas Colaborativos |
| SLR | Systematic Literature Review |
| TAM | Technology Acceptance Model |
| VCS | Version Control System |

# List of Figures

# List of Tables

# Chapter 1
# Introduction

Open Source Software (OSS) projects have risen to great prominence within the last several years (Krogh and Hippel, 2003; Nakagawa et al., 2008). In OSS projects, the source code is licensed to make it freely available to anyone who wishes examine it or change it for their own purposes, redistribute copies and to run the program.

Many OSS projects leverage contributions from geographically distributed volunteers and require a continuous influx of newcomers for their survival, long-term success, and continuity. According to Qureshi and Fang (2011), it is essential to motivate, engage, and retain new developers in a project in order to promote a sustainable number of developers. Furthermore, some studies report that newcomers are a source of innovation for new ideas and work procedures that the group needs (Kraut et al., 2012).

However, newcomers usually face many difficulties when making their first contribution to a project. OSS project newcomers are usually expected to learn about the project on their own (Scacchi, 2002). Dagenais et al. (2010) compare them to explorers in a hostile environment who need to orient themselves. Thus, a major challenge for OSS projects is providing newcomer support.

Previous research related to newcomers' joining process examined the dynamics driving OSS contributors, mostly focusing on the motivations for becoming a member, roadmaps to becoming a core developer, or indicators of potential long-term commitment (Hars and Ou, 2001; Ye and Kishida, 2003; Jergensen et al., 2011; Schilling et al., 2012; Zhou and Mockus, 2012). An understudied aspect of the OSS joining process is what happens during the period after a newcomer decides to participate and before their first code contribution is accepted and included in the shared repository. This period is particularly relevant to OSS projects, as many newcomers do not want to join or remain at the project, only to post a single contribution (e.g., a bug correction or a new feature). What happens in this period affects, for example, students in computer courses whose assignments include OSS project contribution, and professional developers who find a bug or wish to customize a particular software product. During this learning period, newcomers face barriers that can result in their decision to give up contributing. Thus, as Karl Fogel (Fogel, 2013) states, *"if a project doesn't make a good first impression, newcomers may wait a long time before giving it a second chance."*

OSS project newcomers vary, presenting different contribution reasons, backgrounds, and amount of available contribution time. These differences influence the amount of effort newcomers put toward overcoming contribution barriers. According to Kanfer (1990), in the face of obstacles, motivation determines the direction of the individual's behavior, effort level, and persistence level. Therefore, newcomers with stronger motivation (e.g., receiving a grade in a school or being paid to contribute) may fight harder than those users who want to work on issues they found themselves, or who want to return to the community. In addition, developers who have a little time to contribute to an OSS project can give up if they cannot quickly overcome the barriers.

Thus, in this thesis we focus on the barriers newcomers face while trying to make their first OSS project contribution. With a more in-depth understanding of the barriers, researchers and community can invest their efforts in building or improving tools and processes, ultimately gaining more contributions, such as '*drive-by commits*' (Pham et al., 2013), which are small changes made by developers who are only casually or briefly interested in a project and do not intend to have a prolonged engagement.

## 1.1 Research Questions

The goals of this research are to identify and understand the barriers newcomers face when trying to make their first contribution to OSS projects, and to propose and evaluate a portal to help newcomers to overcome some of these barriers. The overall question addressed by this thesis is:

***"How to support newcomers placing their first contribution to Open Source Software projects?"***

To guide our answer to this question, we have defined specific research questions:

RQ1. What barriers do newcomers face when making their first OSS project contribution?

RQ2. How might a tool support newcomers' first contributions?

## 1.2 Research Design and Thesis Organization

We used multiple empirical methods to address this thesis' research problem and answer the research questions. More specifically, this research results from combining systematic literature review, qualitative interview analysis, experimental study comprising qualitative analysis of newcomers' diaries and quantitative analysis of newcomers' self-efficacy and portal usefulness, ease of use, and predicted future use. The research design comprises two phases and complementary studies, as presented in Figure 1.1, and described in the following.

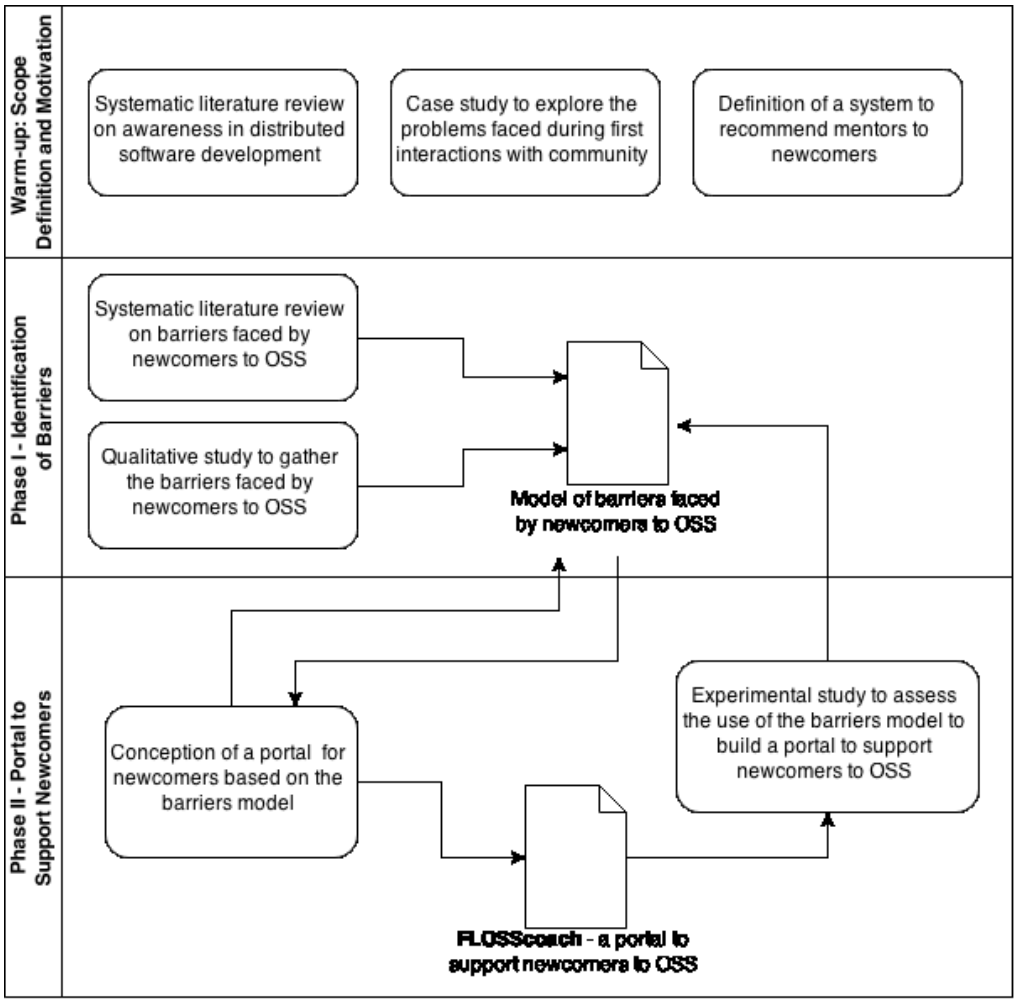



Figure 1.1. Research design.

**Warm up – Scope definition and Motivation.** This phase comprised studies conducted during the definition of this thesis' scope, and helped us to find our research motivation. The first study we conducted was a systematic literature review on the awareness mechanisms for distributed software development (Steinmacher et al., 2010, 2013a). We analyzed 91 primary studies and classified them according to the 3C collaboration model (Fuks et al., 2005, 2007) and to the Gutwin et al. (1996) awareness framework. Coordination was by far the most supported dimension of the 3C model, whereas communication was less explored. We also observed that workspace awareness elements play a central role on distributed software development.

After this first study, we decided to focus on open source software, which is an instance of distributed software development. We then proposed a recommendation system for finding the most appropriate project member to mentor newcomers in a technical task (Steinmacher et al., 2012b). The idealized system makes use of temporal and social aspects of a developer's behavior, and of recent contextual information to recommend a mentor.

Then, we decided to gather more evidence of the research problem we were addressing. We conducted a case study to verify how the first interactions in mailing lists and issue trackers influenced newcomers' decisions to contribute to OSS projects (Steinmacher et al., 2012a, 2013b). We found that less than 20% of the newcomers analyzed became long-term contributors. There was evidence that newcomers' decisions to abandon the project were influenced by the authors of the answers and by the type of answer received. After conducting this study, we defined the scope of this thesis, and started the Phase I studies.

**Phase I – Identification of the Barriers**. This phase comprised two studies to identify barriers faced by newcomers when making their first OSS project contributions, aiming to answer RQ1. First, we empirically gathered the barriers from the literature by means of a Systematic Literature Review focused on the barriers faced by OSS project newcomers. We chose this method because the knowledge about problems newcomers face was spread across the literature, since no previous study specifically focused on this point. The main contribution of this study was aggregating and organizing the barriers evidenced by different studies and creating a model. We then conducted a qualitative analysis over three data sources. The first source was the feedback obtained from students of OSS courses, who were newcomers to OSS projects. The second was answers to an open question sent to several OSS communities. The third source was a set of interviews conducted with OSS project members and newcomers. Afterwards, we analyzed the outcomes obtained from the systematic review together with the qualitative analysis, resulting in a preliminary barriers model, which was used as the input for the following phase of this thesis. For details about the method followed and the resulting model of Phase I, refer to Chapter 3.

**Phase II – A portal to support newcomers.** During this phase, and aiming to answer RQ2, our goal was to apply, evaluate, and evolve the preliminary barriers model in order to support OSS project newcomers' first contributions. In this sense, as detailed in Section 0, we built a portal using information and strategies gathered from practitioners, and organized them according to the preliminary barriers model. To evaluate the portal and model, we conducted an experimental study with students. The evaluation relied on: qualitative analysis of diaries written during students' contributions (detailed in Section 4.2.1); a self-efficacy questionnaire (Bandura, 1977, 1986) (detailed in Section 4.3); and on the Technology Acceptance Model (TAM) (Davis, 1989) (detailed in Section 4.4). The results enabled us to improve the barriers model and the portal according to the feedback received. Chapter 4 provides more details about the method followed and the Phase II results.

This thesis is organized as following: in Chapter 2, we present an overview of the state-of-the-art of newcomers' joining process; in Chapter 3, we describe the Phase II studies, which resulted in the preliminary barriers model; and, in Chapter 4, we detail research Phase II, comprising details of the conception and evaluation of the OSS

newcomer support portal. Finally, in Chapter 5, we present a discussion about our research and, in Chapter 6, the conclusions and future directions.

## 1.3 Scope

In this section, we describe the scope of this research and define some terms. In term of focus, we restrict this thesis to OSS projects instead of general software projects because of their distinct nature. Volunteers easily give up on the project at any moment. By contrast, in an industrial setting, for example, developers are hired and their behavior is bound by a contract; moreover, they are generally supported by mentors, can attend formal training sessions, and are subject to other training and retaining specific strategies adopted by the companies' human resources. Therefore, our claims are not directly generalizable to software development.

In addition, when we refer to "OSS projects," we mean open collaboration (Forte and Lampe, 2013) and community-based open source projects (Capra and Wasserman, 2008). We focus on projects for which the development is typically carried out by volunteers, even tough, projects can be sponsored and some contributors may be employed or hired by a company or an organization that wants to lead the project or accomplish specific tasks. In open collaboration and community-based OSS projects, mailing lists, issue trackers, and source code (in versioning systems) are publicly available. Thus, any skilled person who wants to contribute can access the current code, choose (or report) an issue, address it, and submit a patch.

In our study, we avoid projects with specific and complex domains, including those that deliver development frameworks or scaffolding technologies, like web servers and operating systems. This kind of project demands higher and more specific skills and knowledge, and may involve a specific developer profile. These characteristics can obscure some possible barriers encountered by newcomers, since these newcomers primarily face complex problems related to the domain and specific technologies.

There are different ways a newcomer can begin contributing to an OSS project, including translation, bug triaging, bug reporting, user support, and coding. In this thesis, we solely focus on source code contributions. Therefore, we define *newcomers* as developers (i.e., people with a development background) who want to place their first *code contribution* to a project, and who have no previous experience with that project.

We aim to understand the specific barriers newcomers face during the initial contribution stage, which ranges from the moment that the developers decide to contribute to the project, until the point when the community accepts and integrates their code contribution. As opposed to previous research that studies the whole joining process (detailed in Section 2.3), we are not interested in analyzing the way newcomers are attracted to a project, become project members, or migrate from the project periphery to the core. Rather, we seek to understand the problems that hinder

newcomers from placing their first contribution, regardless of their intention to become core members. In this sense, we propose a model (Steinmacher et al., 2014b) composed of the stages that are common to, and the forces that are influential to, newcomers being drawn in or pushed away from a project. We present these stages and forces in Figure 1.2, which we use to better understand the joining process, present the scope of our research, and further organize the relevant research literature (Section 2.3).

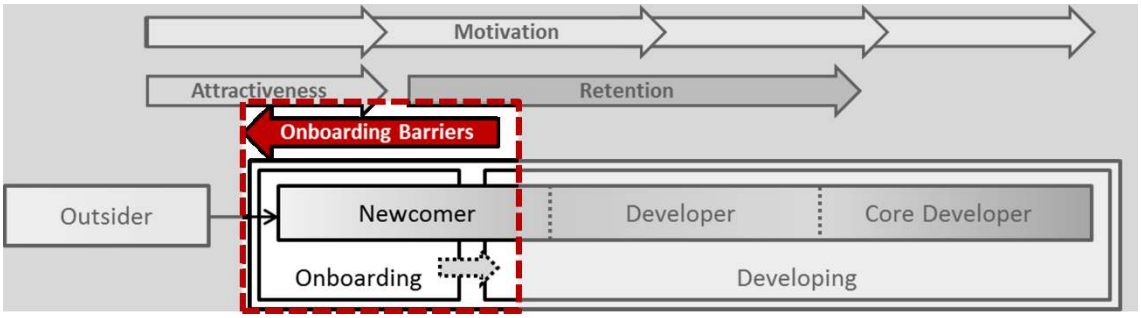



Figure 1.2. Model of developer joining process, presenting the stages and forces at work.

We claim that newcomer contributions require more than just motivating and attracting people. If communities want more contributions, they need to support newcomers to overcome the *barriers* associated with the first contribution, especially because many newcomers only want to place a single contribution. This thesis focuses on this initial stage; both by better understanding the barriers newcomers face when trying to make their first project contributions, and by proposing a portal that supports the newcomers in overcoming these barriers.

## 1.4 Main Contributions

We claim that this thesis contains two main novel contributions, which map onto the research questions presented in Section 1.1, and are related to the artifacts generated in research phases I and II, as presented in Section 1.2. The aforementioned contributions are:

- **Empirical identification and modeling of the barriers faced by newcomers to OSS projects**. Whereas the studies found in the literature focus on developers' activities during the joining process and the steps towards becoming core members, our study offers a unique emphasis on the forces and barriers faced by newcomers to OSS projects when trying to make their first contributions. We conducted studies to further investigate and model these barriers, and we gathered the data to be analyzed from four different sources: (i) a systematic literature review; (ii) feedback from students that contributed to OSS projects; (iii) answers to an open question sent to OSS project developers' mailing lists; and (iv) semi-structured interviews conducted with OSS project dropouts,

newcomers, and members. The main contribution of this phase is a model that represents and categorizes the barriers faced by newcomers.

- **A portal to support newcomers' first contributions to Open Source Software projects**. We applied the barriers model to a practical setting by building a portal to support newcomers' first OSS contribution. We evaluated the portal conducting a study with newcomers to OSS projects, which showed that the newcomers who used the portal felt more comfortable and confident in contributing, and found the portal useful and easy to use.

## 1.5 Other Results

This research resulted in scientific publications, undergraduate diploma theses, and projects funded by national and regional agencies. In the following subsections, we present these results.

### 1.5.1 Published Papers

During the PhD program, we published several papers related to this thesis' topic. So far, the main results were published in papers at OSS 2014, CRIWG 2014, SBES 2014, ACM CSCW 2015, and the Information and Software Technology Journal. The summarized references for the papers originated from this research are:

Papers related to warm-up

**Paper 1.** SILVA, J.T.; WIESE, I.S.; STEINMACHER, I.; GEROSA, M.A. "*MinerAll: Uma ferramenta para extração e mineração de dados de repositórios de software livre.*" In: ***XII Workshop de Software Livre***, 2011.

This paper introduces MinerAll, a tool used to extract data from Apache repositories, which was used to conduct some preliminary studies that are not part of this thesis.

**Paper 2.** STEINMACHER, I.; CHAVES, A.P.; GEROSA, M.A.. "*Awareness Support in Distributed Software Development: A Systematic Review and Mapping of the Literature*". ***Computer Supported Cooperative Work Journal***, v. 22, p. 113-158, 2013.

We systematically analyzed the literature of awareness for distributed software development and classified the studies according to the 3C collaboration model (Fuks et al., 2005) and to the Gutwin et al. (1996) Awareness Framework. This study helped us defining this thesis' initial scope.

**Paper 3.** STEINMACHER, I.; WIESE, I.S.; GEROSA, M.A. "*Recommending mentors to software project newcomers.*" In: ***2012 Third International Workshop on Recommendation Systems for Software Engineering (RSSE)***, 2012. p. 63-67

The barriers faced by PhD students when they tried to contribute to OSS projects motivated this paper. We collected feedback from these students and qualitatively analyzed it. The result of this analysis is now part of the results presented in Chapter 3.

**Paper 4.** STEINMACHER, I.; WIESE, I.S.; CHAVES, A.P.; GEROSA, M.A. "*Newcomers Withdrawal in Open Source Software Projects: Analysis of Hadoop Common Project.*" In: ***2012 Brazilian Symposium on Collaborative Systems (SBSC)***, 2012. 10 pp.

**Paper 5.** STEINMACHER, I.; WIESE, I.S.; CHAVES, A.P.; GEROSA, M.A. "*Why do newcomers abandon open source software projects?*". In: ***2013 6th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)***, 2013. p. 25-32.

Papers 4 and 5 report the results of a study aimed to investigate how first interactions in mailing lists and issue trackers influence newcomers' onboarding. This served to motivate the problem addressed here.

**Paper 6.** STEINMACHER, I.; GEROSA, M.A. "*How to Support Newcomers' Onboarding to Open Source Software Projects.*" In: ***The 10th International Conference on Open Source Systems (OSS'2014), Doctoral Symposium***. 2014. 3pp.

This doctoral symposium paper provides the method proposed for this thesis and discusses the contributions it aimed to bring to the area, as reported in this chapter.

**Paper 7.** STEINMACHER, I.; GEROSA, M.A.; REDMILES, D.F. "*Attracting, Onboarding, and Retaining Newcomer Developers in Open Source Software Projects.*" In: ***Workshop on Global Software Development in a CSCW perspective.*** 2014. 4pp.

This study reports the model composed of forces and stages that are influential to the developers' joining process in OSS projects. This model is used to explain the scope of this thesis in Section 1.3 and to organize the related work in Section 2.3.

<u>Papers related to Phase I – Identification of Barriers</u>

**Paper 8.** STEINMACHER, I.; CONTE, T.U.; GEROSA, M.A.; REDMILES, D.F. "*The Hard Life of Open Source Software Project Newcomers.*" In: ***7th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE 2014)***. 2014.

This paper reports partial results for the qualitative study conducted to identify problems faced by newcomers to OSS. It presents the analysis of data gathered from two sources: feedback obtained from students that joined OSS projects during a course; and answers to an open question sent to OSS communities. Chapter 3 includes this study's results.

**Paper 9.** STEINMACHER, I.; GRACIOTTO SILVA, M.A.; GEROSA, M.A. "*Barriers Faced by Newcomers to Open Source Projects: A Systematic Review*" In: ***10th International Conference on Open Source Systems, OSS 2014.*** 2014. 153–163.

**Paper 10.** STEINMACHER, I.; GRACIOTTO SILVA, M.A.; GEROSA, M.A.; REDMILES, D.F. "*A systematic literature review on the barriers faced by newcomers to open source software projects.*" ***Information and Software Technology***, v. 59, p. 67-85, 2015.

Papers 9 and 10 report the Systematic Literature Review on barriers faced by OSS project newcomers. It is one of the empirical methods used to identify the barriers reported in this thesis. This study is part of the results reported in the Chapter 3.

**Paper 11.** STEINMACHER, I.; CHAVES, A.P.; CONTE, T.U.; GEROSA, M.A. "*Preliminary Empirical identification of barriers faced by newcomers to Open Source Software projects.*" In: ***Proceedings of the 28th Brazilian Symposium on Software Engineering (SBES 2014)***, 2014. p. 51-60.

Paper 11 reports method and the resulting preliminary barriers model obtained from the qualitative analysis of data from different sources, as part of Phase I of this thesis. This study is part of the results reported in the Chapter 3.

**Paper 12.** STEINMACHER, I.; CONTE, T.U.; REDMILES, David F.; GEROSA, M.A. "*Social Barriers Faced by Newcomers Placing Their First Contribution in Open Source Software Projects.*" In: ***18th ACM Conference on Computer-Supported Cooperative Work and Social Computing (CSCW 2015)***. 2014. p. 1-14.

Paper 12 focuses on the social barriers identified during our qualitative study and links them to the literature that investigates newcomers in other open collaboration communities. The results of this study are discussed in Section 3.2.6.

**Paper 13.** STEINMACHER, I.; GEROSA, M.A. "*Choosing an Appropriate Task to Start With in Open Source Software Communities: a Hard Task.*" In: ***Proceedings of the 20th International Conference on Collaboration and Technology, 2014 (CRIWG 2014)***. p. 349-356.

**Paper 14.** STEINMACHER, I.; CONTE, T.U.; GEROSA, M.A. "*Understanding and Supporting the Choice of an Appropriate Task to Start With In Open Source Software Communities.*" In*:* ***Proceedings of 48th Hawaii International Conference on System Sciences (HICSS-48)***. 2015. p. 1-10.

In Paper 13 and Paper 14, we report an in-depth investigation of "*difficulty to find a task to start with.*" We identified reasons and problems related to this barrier, as well as a set of strategies that can support newcomers in overcoming this barrier.

<u>Paper related to Phase II – Portal to Support Newcomers</u>

**Paper 15.** STEINMACHER, I.; WIESE, I.; CONTE, T.U. ; GEROSA, M.A. *"Increasing the self-efficacy of newcomers to Open Source Software projects." In*:

***Proceedings of the 29th Brazilian Symposium on Software Engineering (SBES 2015)***, 2015. p. 160-169.

In Paper 15 focuses on the self-efficacy results that are part of the evaluation of the model of barriers, represented by FLOSScoach, a portal to support newcomers to OSS. This is part of this thesis, presented in Section 4.3.

Other related publications

**Paper 16.** SILVA, J.T.; GEROSA, M.A.; WIESE, I.S.; RÉ, R.; STEINMACHER, I. "*An Extensible Service for Experts Recommendation on Distributed Software Development Projects.*" In: ***2012 Seventh IEEE International Conference on Global Software Engineering Workshops (ICGSEW)***, Porto Alegre. 2012. p. 18.

**Paper 17.** MACIEL, A.C.; STEINMACHER, I.F.; GRACIOTTO SILVA, M.A. "*Padrões de Socialização de Novatos em Projetos de Software Livre.*" In: ***XV Workshop Internacional de Software Livre***, 2014. p. 1-10.

**Paper 18.** SILVA, J.O.; WIESE, I.; STEINMACHER, I.; GEROSA, M.A. *"Pagamento Atrai Colaboradores de Longo Prazo ou Prestadores de Serviço? Uma Investigação Inicial sobre o Google Summer of Code 2014". In: Simpósio Brasileiro de Sistemas Colaborativos (SBSC 2015).*

**Paper 19.** PRUDENCIO, M.V.; COGO, F.R.; CHAVES, A.P.; STEINMACHER, I. *"Participação de Mulheres em Projetos de Software Livre: Caso da Comunidade Brasileira da Mozilla Foundation". In: XVI Workshop de Software Livre (WSL 2015)*, 2015. p. 1-8.

### 1.5.2 Awards and Nominations

Paper 4 was awarded as the 2nd best paper of the *2012 Brazilian Symposium on Collaborative Systems (SBSC).*

Paper 14 was nominated for the best paper award of the "Software Technology" track at the Hawaiian International Conference on Software Systems

### 1.5.3 Funding

**Project 1.** Chamada de Projetos Fundação Araucária nº 05/2011 (2012 – 2014). Análise de redes sociais de desenvolvedores para predição de bugs em projetos de software (*Analysis of developers' social networks to predict bugs in software projects*). Coordinator: Igor Fabio Steinmacher – R$ 8,920.00

**Project 2.** Edital Universal MCT/CNPq 014/2013 (2013 – 2015). Suporte a novatos em projetos de software livre (*Supporting newcomers to open source software projects*), Coordinator: Marco Aurélio Gerosa – R$ 21,000.00

**Project 3.** Edital UTFPR Extensão PROREC 02/2014 (2014-2015). Mulheres Livres: Entrada de Mulheres em Projetos de Software Livre (*Free Women: Supporting Women to join Open Source Software projects*). Coordinator: Igor Fabio Steinmacher – R$ 4,800.00.

### 1.5.4 Advising

**Graduation Thesis 1.** José Teodoro da Silva. Uma arquitetura extensível para localização de especialistas em projetos de desenvolvimento distribuído de software. 2011. Trabalho de Conclusão de Curso - Universidade Tecnológica Federal do Paraná. Advisor: Igor Fabio Steinmacher.

**Graduation Thesis 2.** Ana Claudia Maciel. Padrões de Socialização de Novatos em Projetos de Software Livre. 2012. Trabalho de Conclusão de Curso - Universidade Tecnológica Federal do Paraná. Advisors: Marco Aurélio Graciotto Silva / Igor Fabio Steinmacher.

**Graduation Thesis 3.** Erick William dos Santos. Identificação de especialistas em projetos de software livre baseada em suas contribuições utilizando análise temporal. Início: 2012. Trabalho de Conclusão de Curso - Universidade Tecnológica Federal do Paraná Advisors: Marco Aurélio Graciotto Silva / Igor Fabio Steinmacher.

### 1.5.5 Software Developed

**MinerAll** – a tool that aims at mining OSS repository data. This tool enables investigation of developer collaboration and the degree of artifact dependency. Advantageously, this tool is flexible, allowing the design and coupling of modules to mine different types of Version Control Systems, and also storage in different media (databases, file systems, etc.). The tool is available under General Public License (GPL) v3 at http://minerall.googlecode.com.

**FLOSScoach** – the portal developed as the product of our research's Phase II. It is publicly available to use at http://www.flosscoach.com, with a few projects mapped. For each project, we have organized existing information and strategies to support newcomers in overcoming barriers.

**JabRef**[1] – the motivation of this thesis came from a course in which the author developed some new features for JabRef, a bibliography manager written in Java[2]. The author contributes occasionally to this project and is currently a member of the project.

### 1.5.6 Serving the community

Because of the results of this research and its repercussion, the author was invited to serve the community as organization chair of Workshop de Software Livre (Free Software Workshop) (WSL 2014 and WSL 2015) and to give an invited talk at Forum Goiano de Software Livre - FGSL 2014 (Goiás Free Software Forum).

[1] http://jabref.sourceforge.net/
[2] http://www.java.com/

# Chapter 2
# Related Work

A "newcomer" describes a person who has recently arrived, or is starting to participate in something ("The Free Dictionary," 2014). A newcomer to any group or community faces many barriers before feeling a part of that community, and some of these barriers lead to newcomers' withdrawal. In this chapter, we present an overview of the existing literature on newcomers and their joining process.

## 2.1 Newcomers' Barriers

Newcomers face different kinds of barriers when they try to enter communities. In the sixties, Thoma and Lindemann (1961) reported barriers faced by newcomers to a suburban community of the United States. The authors evidenced that when newcomer families differed from the established norms (occupationally, ethnically, educationally, or in their behavior), some were forced to leave the town to protect themselves. Also in the sixties, Lofland and Lejune (1960) studied newcomers to Alcoholics Anonymous (A.A.), and analyzed their socialization with existing members. They hypothesized that if the newcomer and the A.A. group displayed similar social class presentations, the initial socialization would be higher. However, the hypothesis was not supported for the analyzed sample, because the more different the newcomer and group social class were, the easier it was for the newcomer to stand out and socialize.

Foner and Alba (2008) analyzed the issues caused by newcomers' religion to join local Western European societies. Three barriers were reported as critical: the religious background of immigrants; the native population's religiosity; and historically-rooted relations and arrangements between the state and religious groups.

Baier (2005) studied the problems faced by international students of the University of Michigan, who were newcomers to the United States. They assessed the impact of culture, language, religion, personality, and social support during their adaptation. The study revealed that non-western students had a more difficult time than western students, and women were found to have more problems adjusting than men. They also showed that international students with English as one of their primary languages had stronger support networks.

In this thesis, we are interested in newcomers to Open Source Software projects, which are online open collaboration communities (Forte and Lampe, 2013). Choi et al. (2010) claim that, although online groups are similar to offline ones in many dimensions, one cannot directly generalize the findings from offline groups to online groups. The explanation is related to the nature of the communities and the motivation behind them.

Open collaboration communities count on volunteer contributions, which makes it easier for the members to leave their groups (Burke et al., 2009) because volunteer members generally are not bound to their groups through an employment contract (Choi et al., 2010; Choi, 2012). Moreover, the impoverished awareness information, lack of trust, and relatively weak interpersonal ties between members in many online groups make it more problematic to attract and retain people than in face-to-face groups (Tidwell and Walther, 2002).

Because of the aforementioned differences, group commitment is often lower than in conventional offline groups, and it becomes easy for volunteers to leave. This poses additional challenges in encouraging productivity and discouraging withdrawal (Wang et al., 2012).

Studying newcomers and the problems they face in online open collaboration communities is a contemporary problem that still needs to be further investigated. According to Zhu et al. (2013) one of the most significant challenges for many online communities is increasing members' contributions over time. In this sense, the next sections aim to present the state-of-the-art of newcomers' joining process to online communities (Section 2.2) and, more specifically, for newcomers to Open Source Software (in Section 2.3).

## 2.2 Newcomers to Online Communities

The internet, or Web, evolved to become a platform for collaboration, sharing, innovation, and user-created content, the so-called Web 2.0 (O'Reilly, 2005). For instance, online social networks and online communities emerged as Web 2.0 phenomena. Online communities are online spaces wherein a group of individuals comes together around a common interest, and, without time and space constraints, shares among themselves self-generated content (Preece, 2001; Jin et al., 2010).

Many online communities are "open collaboration," meaning they are maintained by volunteer contributions. Communities' viability and success thus depend on their members' active participation and content contribution, as well as the sustainable community registration of newcomers (Lee et al., 2014). This necessary challenge of continually attracting new collaborators to join and engaging them to participate has motivated researchers to study the online communities' joining process.

For instance, Burke et al. (2009) analyzed Facebook[3] sharing behavior, finding that newcomers who initially see their friends contributing in turn share more content themselves. Additionally, those newcomers who began their participation with a contribution, received feedback, and garnered a wide audience were more likely to share additional content. Thus, learning by example and receiving proper feedback seems to significantly influence on future contributions.

To propose and test a model that outlines newcomers' behavior and psychological mediating and moderating effects on participation, Tsai and Pai (2014) studied newcomers to virtual communities hosted by a large Taiwanese Internet provider. They conducted a two-phase survey, followed by an analysis of newcomers' data extracted from the virtual communities. They found that the combined fulfillment of needs for autonomy, relatedness, and competence determines newcomers' cognitive social identity, which then influences participation behavior and results in commitment to the community and collective self-esteem.

Fugelstad et al. (2012) conducted a study with MovieLens[4] newcomers to determine the factors that predict long-term contribution. They surveyed nearly 4000 new users of the MovieLens and logged their usage history, finding that motivations for volunteering, pro-social behavioral history, and other community-specific motivations explained the amount of use and engagement with some activities. Hsieh et al. (2013) similarly found that pro-social behavior and community-specific motivation are long-term contribution predictors in reddit[5]. They also evidenced that volunteers who had received help from other users were more likely themselves to support newcomers.

In a different context, Sande (2013) studied an online mathematics help forum's newcomer joining process. They paid five helpers to join the community for eight weeks. Afterwards, they analyzed data from the forums, posts, and interviews conducted with the helpers. They analyzed the progression of one of these five members (the most successful) and found that the joining process involved cultivating a sense of belonging and relationship building, creating a voice, and demonstrating community commitment.

In addition to the studies presented above, the literature is dominated by studies of newcomers joining Wikipedia.[6] Recently, Wikipedia has suffered from a reduction of new contributors, in what Halfaker et al. (2013a) called "the decline of Wikipedia." This, in addition to the relatively low return rate of Wikipedia contributors (with only 0.0002% of users who make one edit becoming regular editors) (Panciera et al., 2009), motivates much of the Wikipedia newcomer research.

[3] http://www.facebook.org
[4] http://movielens.org
[5] http://www.reddit.com
[6] http://www.wikipedia.org

Bryant et al. (2005) analyzed how Wikipedia users' motivations and role perceptions changed as they became more engaged. They conducted interviews with editors and qualitatively analyzed how newcomers started contributing, and how they became frequent editors. They found that newcomers tend to make only minor contributions, which spring from users' personal knowledge and relate to domains in which they feel comfortable and competent, such as hobbies and personal interests. Moreover, they reported that newcomers were neither aware of the community they were about to contribute to, nor that there were rules to follow.

Lampe et al. (2012) reported the results of a study conducted with 185 students from 22 U.S. universities who, as part of their coursework, contribute content to Wikipedia articles. After the course, they were surveyed about their participation and their intention to participate in Wikipedia in the future. They found that necessary language (English) familiarity and technical skills affected their degree of engagement. Overall, the retention rate 2 months after the course was about 4.06%, which is better than the 0.0002% presented by Panciera et al. (2009). This higher rate demonstrates that course assignments can motivate long-term editors.

Farzan and Kraut (2013) qualitatively and quantitatively studied the participation of 640 psychology graduate and undergraduate students who contributed to Wikipedia, reporting the "*benefits of an authentic writing experience that would be read by thousands of people*." Yet, during the interviews with the students, many of them reported demotivation due to hostile experienced editor behavior. For instance, some students complained about reversions and deletions that occurred without proper/polite explanation. The authors summarized the issues by highlighting the importance of positively receiving newcomers and incorporating them into the community.

Zhu et al. (2013) conducted a field experiment studying feedback's impact on Wikipedia contributions. One goal was to analyze the influence on newcomers of positive, negative, directive, and social feedback. The authors sent feedback messages to new editors who had recently created articles, and analyzed the feedback's effect. Their results suggest that (mildly) negative feedback prompted newcomers to work harder on the target article, and had no general motivation effects.

Vora and Komura (2010) analyzed the impact of newcomers' joining experience on editing Wikipedia pages, interviewing users who were willing to, but ultimately did not, contribute, as well as new editors who had made fewer than 5 contributions. They mainly reported issues with learning how to use the wiki editor, which prompted some newcomers to give up. The result of this research was an unevaluated prototype aimed at overcoming the technical issues interviewees reported.

Halfaker et al. (2011) analyzed 400,000 Wikipedia revisions to understand reverts' effects on editors. Editors use reverts to fix mistakes, repair vandalism, and help guarantee content quality. The authors aimed to understand the effect that reverts have on contributors' demotivation. They found that being reverted decreases both an editor's

probability to continue editing, as well as the motivation of editors who do indeed continue editing. Furthermore, they analyzed how this affects newcomers in particular, finding that they are less likely than old-timers to continue editing after being reverted, and, if they do continue to edit, are subsequently less active. These results are complemented by the historical analysis on reverts made by Suh et al. (2009), who found that an excessive number of reverts on newcomers evidenced the Wikipedia community's growing resistance to new content, especially from occasional and new editors.

Motivated by the increasing number of reverts, Halfaker et al. (2013a) analyzed history logs to check if increasing rejection rates have caused a decrease in the retention of desirable newcomers. They showed that newcomers' quality has not substantially decreased, but rejection of desirable newcomers' contributions rose to become a significant, negative retention predictor. They blamed new quality and efficiency policies, in which newcomers were rudely greeted by automated quality control systems and were overwhelmed by the rule system's complexity. Their conclusion was that "Wikipedia has changed from the encyclopedia that anyone can edit to the encyclopedia that anyone who understands the norms, socializes himself or herself, dodges the impersonal wall of semi-automated rejection, and still wants to voluntarily contribute his or her time and energy can edit."

From a more positive angle, Choi et al. (2010) identified socialization tactics frequently used in Wikipedia, and examined their influence on newcomers' engagement. To identify the socialization tactics, they qualitative analyzed 12 WikiProjects' edit histories, finding that welcome messages, assistance, and constructive criticism appear to have retarded the natural decline in newcomers' editing with time. However, invitations were associated with a faster drop-off in edits.

Morgan et al. (2013) presented "Teahouse," a new editor space that supports newcomers' joining, consisting of a relatively simple set of tools, norms, and procedures. Teahouse was presented as a safe place where new editors could have their questions answered by friendly Wikipedians. The authors conducted a pilot study to evaluate the effectiveness of the tool. They surveyed Teahouse users and compared Teahouse versus regular newcomers' subsequent activities. Their results suggest that Teahouse satisfaction was high, and that newcomers joining using the tool generally made more edits and participated in more discussions.

Halfaker et al. (2013b) presented another newcomer support tool; they evaluated the Article Feedback Tool, a system designed to elicit lightweight contributions from Wikipedia's readers. Users could use the tool to rate their experience, provide four different types of feedback about the article (suggestion, praise, problem or question), and submit a contribution. The researchers conducted experiments in live English Wikipedia and found that simple, straightforward, and prominent contribution types, like comments and feedback, increased participation without sacrificing quality.

Faulkner et al. (2012) evaluated the effectiveness in preserving the contribution rate of minor changes to semi-automated warning messages sent to newcomers by a Wikipedia quality control tool. They evaluated the impact of two types of changes in warning message templates: (i) a more personalized message, with a more informal, passive, and less accusatory tone; and (ii), a shortened message, with simpler, more understandable warning language. Assessing the changes' effectiveness in Wikipedia experiments, they found that modifying first time warnings significantly increased the likelihood that a newcomer editor would, in the short-term, contribute more.

These studies relate to our research, but they study newcomers to online communities in general. In Table 2.1 we summarize these studies, showing that Wikipedia is the dominantly explored community, and that there is a balance between use of quantitative and qualitative analysis. In the next section, we present the studies that specifically deal with newcomers to OSS projects.

Table 2.1. Summary of studies regarding newcomers in online communities.

| Study | Method | Virtual Community | Main Findings |
|---|---|---|---|
| (Bryant et al., 2005) | Qualitative analysis on Interviews | Wikipedia | Newcomers usually start adding value to a familiar subject; they are not aware they are part of a community and that there are a set of rules to follow. |
| (Suh et al., 2009) | Quantitative analysis on historical data | Wikipedia | Amount of reverts increased and the most affected users are the new and occasional editors |
| (Choi et al., 2010) | Qualitative/quantitative analysis on historical data | Wikipedia | Welcome message, assistance, and constructive criticism appear to have retarded the natural decline in newcomers' editing with time. Invitations were associated with a faster drop-off in edits. |
| (Vora and Komura, 2010) | Qualitative analysis on interviews with newcomers | Wikipedia | Problems learning the wiki language and using the editor hinder newcomers contributions |
| (Halfaker et al., 2011) | Quantitative analysis on history logs (400,000 revisions) | Wikipedia | Reversions affect newcomers decision to leave |
| (Lampe et al., 2012) | Survey with students joining Wikipedia | Wikipedia | Language and technical skills are influencers of engagement. |
| (Farzan and Kraut, 2013) | Quantitative and qualitative study using survey and interviews conducted with newcomers | Wikipedia | Bad reception and reversions demotivate newcomers |
| (Faulkner et al., 2012) | Experiment on how change warning message tone affect newcomers retention | Wikipedia | Personalized and less rude messages increase the retention. |
| (Halfaker et al., 2013b) | Qualitative and quantitative analysis of the use of a tool to help attracting and motivating new contributions from users | Wikipedia | Participation can be increased by making the means of contribution more prominent with simple and straightforward contribution types like comments and feedback. |
| (Morgan et al., 2013) | Quantitative analysis on contributions of newcomers who succeeded using a tool to support socialization (Teahouse) | Wikipedia | Users of the tool were better succeeded than regular newcomers in terms of amount of edits, amount of articles edited and discussions. |
| (Zhu et al., 2013) | Field Experiment | Wikipedia | Receiving feedback stimulates the newcomers to improve their contributions |
| (Burke et al., 2009) | Quantitative analysis of Facebook newcomers log followed by interviews to confirm | Facebook | Learning from friends (lurking) and receiving feedback and great audience are predictors of future contributions |
| (Tsai and Pai, 2014) | 2 phase method: (i) a priori surveys to find the antecedents; and (ii) quantitative analysis of data extracted from virtual community | Virtual communities from Taiwan | Autonomy, relatedness, and competence determines newcomers' cognitive social identity, which then influences participation behavior through affective commitment and collective self-esteem |
| (Fugelstad et al., 2012) | Surveys and quantitative analysis on the history of 4000 newcomers | Movielens | General volunteer motivation, pro-social behavioral history, and community-specific motivation are good predictors of the amount of use and specific types of activities users engaged in after joining the community. |
| (Sande, 2013) | Qualitative analysis on logs and interviews conducted with hired helpers | Mathematics online helpers community | Membership, influence and immersion are necessary to become central |
| (Hsieh et al., 2013) | Surveys | Reddit | Social-identification with the community, users who were helped by other members and people with pro-social values are more likely to support newcomers. |

## 2.3 Newcomers to Open Source Software Projects

The problems newcomers face when joining software teams motivated studies in both closed source settings and OSS settings. Begel and Simon (2008), Berlin (1992) and Sim and Holt (1998) studied newcomers' joining process in commercial software companies, finding that formal mentoring and working on small tasks helped newcomers find their way into projects. Also in closed source settings, Dagenais et al. (2010) performed a Grounded Theory study with 18 newcomers and identified barriers influencing their integration experience and three primary strategies used to overcome the barriers: early experimentation, internalizing structures and cultures, and progress validation. Their main suggestion was to redesign the projects' landscapes to ease newcomer integration. From these studies in commercial settings, we can see that newcomers face joining barriers in those settings as well. However, as briefly discussed above, OSS projects present different challenges than traditional commercial projects.

As highlighted in the Introduction, we claim that joining an OSS project is a complex process, presented in the model depicted in Figure 1.2 (Section 1.3). This model is composed of the stages common to, and forces influential to, the joining process. The model's central elements are the stages that developers go through; OSS communities should variously invest in these stages to encourage more developers to contribute. An *outsider* represents a potential contributor to the project who is not yet involved with the development. *Newcomers* are people trying to make their first code contributions to the project. *Developers* already contribute to the project, but are not recognized as members and do not have commit privileges. *Core developers* are people recognized by the community as formal contributors or those presenting commit privileges to the repository.

We represent four different forces in the model that influence the progress from one stage to the following. Motivation and project attractiveness are the triggers that push the outsider to contribute. *Motivation* forces represent internal (e.g., learning, self-marketing, and recognition) and external (e.g., scholarship, course assignment, and feature need) motives that drive a developer to join (and keep contributing) to a project. Hence, motivation forces are present in the whole joining model, since lack of motivation leads to drop offs. Significantly, these motivation forces can change/evolve during the development process. For example, some developers onboard a project because of a short-term scholarship, such as the one given by Google Summer of Code, or university course credit, and afterward, in order to learn and self-promote themselves, remain contributors.

*Attractiveness* forces represent the characteristics and actions that the project presents for recruiting new users and developers (Santos et al., 2013). These forces can include type of license, project visibility, project age, and number of developers. Attractiveness and motivation work together to pull outsiders towards the projects. In some cases, attractiveness forces play a special role, pulling in motivated developers who otherwise could not decide which project to support.

The transition from outsider to newcomer occurs when a developer decides to contribute to a project and starts onboarding. During this stage, motivation continues compelling the developer towards the project. However, some opposite forces, which we call onboarding barriers, can hamper this joining process. These forces comprise technical and non-technical barriers, including learning curve, lack of community support, and difficulties figuring out how to start.

Since these barriers can be powerful enough to lead developers to give up contributing to the project (Steinmacher et al., 2013b), understanding how to deal with them is critical to the joining process. An important observation is that these barriers influence both developers willing to make a single contribution as well as those willing to climb higher and become a project member.

On the other hand, retention forces may help push newcomers to remain willing to contribute. *Retention* forces represent the characteristics and actions that a project presents to attract and keep developers. Some of these forces include support initiatives to help newcomers overcome barriers (such as providing tools to facilitate code understanding, or indicating appropriate tasks or pieces of code to start with). Other forces represent mechanisms to support existent developers to contribute more, which triggers, in some cases, a motivation change (e.g., granting commit rights to a developer, or using *gamification* elements).

In the following subsections, we present the state-of-the-art OSS project developers joining process models from the perspective of the forces represented in the proposed model (Figure 1.2).

### 2.3.1 Joining process

Some studies report scripts and cases of successful developers joining OSS projects, presenting this data as a joining process. Briefly introduced in Chapter 1, the onion model (Nakakoji et al., 2002; Ye and Kishida, 2003) is a general layered structure that organizes OSS member roles. Nakakoji et al. (2002) and Ye and Kishida (2003) studied four different OSS projects and found that each member may have one of the eight roles presented in the Figure 2.1. Furthermore, they claim that the onion's layers represent the path that a user ideally runs through to arrive at the project's core. Although they recognize that "*not all members want to and will become core member*s," they create a process a developer needs to follow to contribute to a project.

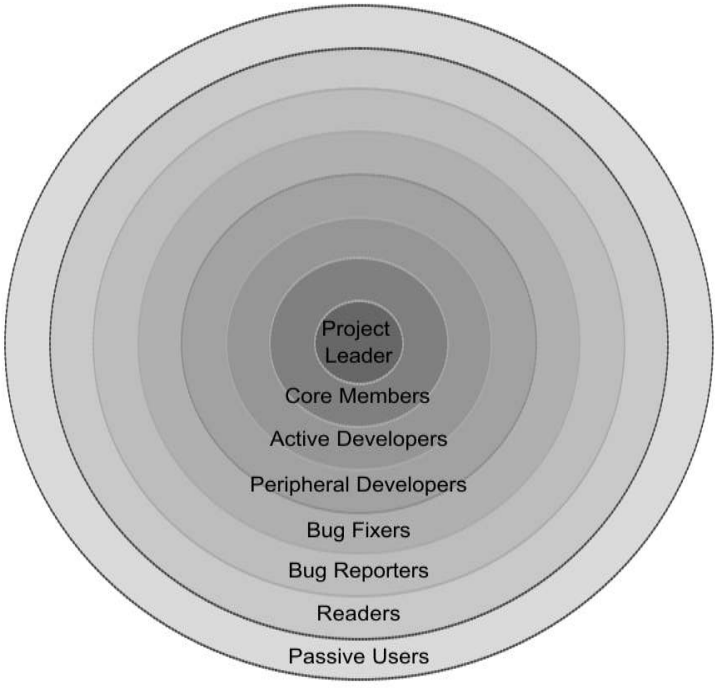



Figure 2.1. Onion Model (Nakakoji et al., 2002).

Jergensen et al. (2011) studied whether the onion model still holds true in large project ecosystems, and how the model might change in such settings. They analyzed data from the GNOME[7] project from 1997 to 2007 and found little evidence for the onion model's description of a gradual participation and socialization process whereby individuals migrate from the project's edges to its core. When considering the contributions in terms of ecosystems, the onion model was supported slightly more often. Herraiz et al. (2006) also analyzed the onion model, studying seven years of more than 1,000 developers' GNOME histories. They found that at least 52.5% of the developers committed before sending a message to the mailing list. Thus, the population analyzed does not comply with the onion model. By further analyzing the history of 20 developers, they found two groups with clearly different joining patterns. They related those patterns to the different behaviors of volunteer and hired developers; whereby volunteers tend to follow a step-by-step joining process, hired developers usually quickly integrate.

Besides the onion model, other studies have sought to map the OSS project joining process. Krogh et al. (2003), for example, proposed the concept of "joining script" after analyzing interviews with developers, emails, source code repository, and documents from FreeNet[8] project data. The joining script describes the level and type of activity a newcomer goes through to become a community member. In the same sense, Ducheneaut (2005) analyzed the Python[9] project's mailing list archives using an approach he called

[7] http://gnome.org
[8] http://freenetproject.org
[9] http://python.org

"computer-aided ethnography." He conducted an in-depth analysis of one successful Python newcomer's socialization history. Based on this individual, the author identified a set of socialization activities that contributed to his success, as well as highlighted the size of the project and the existence of specific rites of passage during his trajectory.

Studies on the developer joining process use different terminologies to identify the stages or roles through which developers need to pass when joining a project. In Table 2.2, we present the different terms used in three studies relative to the names we used.

Table 2.2. Terminology adopted by other studies and in this thesis.

| Krogh et al. (2003) | **Onion Model** | Ducheneaut (2005) | ***Naming used in this thesis*** |
|---|---|---|---|
| N/A | Passive User | N/A | *Outsider* |
| Lurker | Reader | Peripheral Participant | |
| Participant | Bug Reporter | Debbuger/Patcher | |
| Joiner | Bug Fixer | | *Newcomer* |
| Newcomer | Peripheral Developer | | |
| Developer | Active Developer | Developer | *Developer* |
| Core Developer | Core Member | Core Developer | *Core Developer* |
| | Project Leader | | |

Existing literature usually presents the joining process as a set of steps and activities newcomers must perform to become a project member. Moreover, these studies focus on "how to become a core member." We argue that, due to OSS projects' voluntary and collaborative nature, such a stepped model is not the best way to map developers joining process. Our approach differs from the previous ones insofar as we focus on the forces, rather than only the steps and activities, that influence the developers' onboarding and contribution.

### 2.3.2 Motivation

"Motivation" represents the forces driving developers to contribute, including motives behind first and ongoing project contributions.

Lakhani and Wolf (2005) studied individuals' OSS project contribution motivations, surveying 684 OSS developers to find that external motivational factors (in the form of extrinsic benefits such as career advancement) are primary motivators. They also reported that personal enjoyment, challenges derived from writing code, and improved programming skills are top contribution motivators. Hars and Ou (2001) also used a survey to understand the internal and external motives driving developers to participate in OSS projects. Somewhat different from Lakhani and Wolf's study, they found that internal motivators (such as intrinsic motivation, having fun, and community identification) play a role, but that external factors carry greater weight. They highlight building human capital and personal needs for a software solution as primary external motivations.

Hertel et al. (2003) reported the results of a survey conducted with Linux kernel[10] developers. The authors used sociology models to explain social movement and virtual team participation. They found that engagement was determined by developers' identification as a Linux developer, improving their own software and career advantages, and by time lost due to Linux-related tasks.

Shah (2006) conducted a broader study aimed at understanding the developers motivation by analyzing interviews, mailing list archives, and documentation of two OSS projects to understand the motivation of developers. Instead of locating primary motivational forces, they classified contributors into two types: need-driven and hobbyists. Need-driven participants' motives vary, comprising a specific feature's necessity, project reciprocity, desire to integrate their own source code, career concerns, and receiving feedback and improvements on their solutions. Hobbyists, on the other hand, contribute for fun and entertainment.

These are only a few of the studies dedicated to researching motivation in OSS, of which there are many more, including some on intrinsic motivation and extrinsic motivation factors (Bonaccorsi and Rossi, 2004; Roberts et al., 2006; Jergensen, 2007; David and Shapiro, 2008; Oreg and Nov, 2008; Ke and Zhang, 2010; Krogh et al., 2012). Moreover, in general motivation is an extensively studied topic in software engineering (Beecham et al., 2008; França and Silva, 2009, 2010; Sharp et al., 2009; Yu and Ming, 2009; França et al., 2011, 2014b). Indeed, Beecham et al. (2008) reported a systematic review of 92 such studies until 2006. França et al. (2011) updated this systematic review and reported the analysis of 53 additional studies conducted from 2006 to 2010. Both reviews included OSS-related studies in their analysis.

### 2.3.3 Attractiveness

Attractiveness represents forces that projects use to foster contributions from new developers. Studies of strategies and policies to attract volunteers to OSS projects are important, since "*the success of a project is often related to the number of developers it can attract*" (Capiluppi and Michlmayr, 2007). In this section, we report on the studies concerning OSS project attractiveness.

Santos et al. (2013) defined a theoretical cause-effect model for attractiveness as a crucial OSS project construct, proposing its typical causes, (license type, intended audience, type of project, development status), indicators (hits, downloads, members), and consequences (number of open tasks, time for task completion). They tested the model with data from more than 4000 projects and found that the project directly influences the attractiveness. One of their findings is that projects for end-users and

[10] https://www.kernel.org

developers have higher attractiveness. They also found that application domain impacts attractiveness, and they reported that projects listed as multimedia, printing, security and system (application domain) are more attractive, whereas those in database, education, science, and sociology are less attractive. Moreover, they reported that mature projects are more attractive, and that projects licensed under the most common and restrictive licenses tend to be less active and less effective than projects that do not adopt General Public License (GPL).

The findings regarding license restrictiveness presented by Santos et al. are in line with the results presented by Stewart et al. (2006). However, they contradict the results of Colazo and Fang's (2009), which analyzed 62 projects from SourceForge[11] and found that restrictively licensed projects are more attractive to volunteer OSS developers. In another study, Santos et al. (2011) evaluated the impact on attractiveness of OSS project licensing changes. They found that changes from GPL to other licenses positively influenced projects' attractiveness.

Meirelles et al. (2010) built upon Santos' model, inserting source code as a typical attractiveness cause. They observed the influence of structural complexity and software size (lines of code and number of modules), indicating that structural complexity negatively influences attractiveness, whereas software size positively influences it.

Chengalur-Smith et al. (2010) analyzed whether codebase size, project age, and niche size (a measure borrowed from ecology) influenced project attractiveness, finding that these three characteristics indeed influence the project's ability to attract and retain developers.

Ververs et al. (2011) mapped the influential factors that determine developers' participation in the Debian[12] project. They analyzed potentially influential project events and commits over 11 years and found that the highest influences were specific events, such as CeBIT,[13] Debian Day, new or frozen releases, incidents, dependency issues, and the introduction of new developer services.

### 2.3.4 Onboarding Barriers

"Barriers" are the forces that present obstacles to newcomers and contributors who are willing to place their project contributions. These forces can delay or even forego contribution.

Although the current literature focuses on motivation and forces that lead developers to the project's core, these studies neglect those newcomers who do not

[11] http://www.sourceforge.net
[12] http://www.debian.org
[13] http://www.cebit.de

envision a long-term engagement, as well as those who only want to place a single contribution. Counterexamples are Hannebauer et al. (2014), Steinmacher et al. (2014a), Steinmacher et al. (2014d), Steinmacher et al. (2015a) and Steinmacher et al. (2015b), which explicitly focus on studying barriers that influence newcomers' first contribution.

Jensen et al. (2011) analyzed mailing lists of OSS projects to verify if the emails sent by newcomers were quickly answered, and if gender and nationality influenced the kind of answer received. They found that receiving timely responses was positively correlated with future participation. They found few rude replies to newcomers, but reported that this flaming behavior can have a chilling effect, since the mailing list is public. Similarly, Steinmacher et al. (2013b) analyzed mailing lists and issue trackers to study receptivity's influence on newcomer onboarding. Among the factors that influence the decision to abandon, they found evidence that receiving inadequate answers, and the respondent's experience, affects the decision of newcomers. (Tsay et al., 2014) quantitatively analyzed the association of technical and social measures with the likelihood of contribution acceptance. They found that information signaling both good technical contribution practices for a pull request and strong social connection influence the acceptance decision.

Exploring the technical side of projects, Midha et al. (2010) found that an increase in cognitive complexity decreases the amount of contributions newcomers placed. Thus, code complexity was evidenced as an onboarding barrier. Also regarding technical barriers, Wolff-Marting et al. (2013) proposed two patterns to support newcomers in overcoming onboarding barriers. The first pattern aims at helping newcomers build project source code. The second pattern encourages newcomers to submit their modifications to the main development branch by reducing the fear of introducing a new bug. However, they did not report any evidence of the problems they are addressing.

Several other studies proposed ways to help newcomers' make their first contributions; for instance, Cubranic et al. (2005) presented Hipikat, a tool that builds a group memory comprising source code, email discussions, and bug trackers. The tool enables newcomers to request recommendations based on existent artifacts. With a similar goal in mind, Wang and Sarma (2011) presented a tool to enable newcomers to identify bugs of interest and resources related to that bug, as well as to visually and interactively explore the bug's appropriate socio-technical dependencies. Park and Jensen (2009) showed that the use of visualization tools supported the first steps of OSS project newcomers. They found that this kind of tool helps newcomers more quickly find artifacts, and become accustomed to the architecture and code structure.

Canfora et al. (2012) proposed and evaluated an approach to identifying and recommending mentors for open source project newcomers by mining data from mailing lists and source code versioning systems. Steinmacher et al. (2012b) propose a recommendation approach to help newcomers finding the most appropriate project member to mentor a specific technical task. Fagerholm et al. (2014) conducted a case study to assess the impact of mentoring support on developers, finding that mentoring

had a significant impact on newcomer onboarding, allowing them to become more active. These studies presented tools and strategies to help OSS project newcomers. However, they do not provide or use empirical evidences of the barriers their approaches are trying to counteract via albeit useful newcomer support.

### 2.3.5 Retention

"Retention" forces represent the ability and characteristics that a project presents to support newcomers' onboarding and to sustain developer contribution.

Schilling et al. (2012) used the concepts of Person-Job fit and Person-Team fit to evaluate the retention of former Google Summer of Code (GSoC)[14] students in the KDE[15] project. They found that the development experience, alongside with students' familiarity with project coordination practices, were strongly associated with retention. They also report that students do not remain considerably longer if they have underused abilities (in the project) or higher academic education.

Zhou and Mockus (2012, 2015) worked on identifying the newcomers who are more likely to remain project contributors in order to offer active support for them to do so. They found that the interaction between individual l's attitude and the project's climate are associated with the odds that an individual becomes a valuable contributor.

Fang and Neufeld (2009) built upon Legitimate Peripheral Participation (LPP) theory (Lave and Wenger, 1991) to understand developers' motivations to continue contributing in a sustainable way. Results from qualitative analyses revealed that initial participation conditions did not effectively predict long-term participation, but that situated learning and identity construction behaviors were positively linked to sustained participation.

Finally, Qureshi and Fang (2011) used social resource theory to analyze the trajectories of 133 newcomers in 40 projects, identifying distinct classes of newcomer behavior after considering their initial amount and subsequent growth of interactions with core members.

## 2.4 Final Considerations

In this section, we presented studies that deal with newcomers' joining processes in different contexts, including online and offline communities. In the OSS domain, we systematized the knowledge by proposing a *developers' joining model* (Figure 1.2), which

[14] http://developers.google.com/open-source/soc/
[15] http://www.kde.org

represent the stages that are common to and the forces that are influential to newcomers being drawn into or pushed away from an OSS project. As with every model, ours is a simplification and abstraction of reality. Nevertheless, we claim that it is useful for OSS communities' plans to invest in successfully supporting their newcomers.

By presenting the opposing forces in the proposed model, we aim to highlight onboarding barriers' effects. We noticed a lack of studies focusing on the forces that hinder newcomers from contributing to OSS projects. Some studies cite barriers that influence newcomers' overall experiences, however they do not provide an in-depth understanding of the barriers, their relations, and their relevance in multiple projects. Thus, knowledge about contribution barriers is spread thin across the literature, and there is little exploration of the barriers faced by newcomers who ultimately stop contributing or do not become members. To understand these barriers, we conducted an in-depth qualitative analysis on data gathered from OSS project community members and newcomers; our study's qualitative analysis thereby complements the existing literature, which relies mostly on quantitative evidence.

We also suggest that theories from management sciences could be used to support newcomer joining process research. One of these theories is the Legitimate Peripheral Participation (LPP), which explains how participation, situated learning, and identity construction interrelate and coevolve as an individual engages in a community of practice (Lave and Wenger, 1991). This model served as the basis for several studies presented in this chapter (Ye and Kishida, 2003; Fang and Neufeld, 2009).

In addition to LPP, other theoretical models were used by existing studies on joining process: organizational socialization, by Van Maanen and Schein (1977) (Choi et al., 2010; Kraut et al., 2012); group socialization model, by Levine and Moreland (1994) (Burke et al., 2009); person-environment fit concepts (Schilling et al., 2012).

As our review of prior literature shows, our phenomenon of interest – the barriers faced by newcomers to OSS projects – is poorly studied. We found no effort to collect, organize, and/or understand these barriers. Therefore, instead of using a theoretical model, we decided to proceed with a data-grounded, exploratory study (Yin, 2008), aiming to generate preliminary theoretical constructs about the phenomenon based on empirical observations (Smolander et al., 2008).

In the following chapter, we present our exploratory study dedicated to identifying and organizing the barriers OSS project newcomers face.

# Chapter 3
# Barriers Faced by Newcomers to OSS

To identify and organize the barriers faced by OSS project newcomers, we conducted a qualitative study relying on four different sources: (i) studies gathered via a systematic literature review (SLR) that aimed at identifying and organizing the barriers evidenced by the literature (Steinmacher et al., 2014c, 2015c); (ii) feedback from 9 students after they contributed to OSS projects (Steinmacher et al., 2014d); (iii) 24 answers to an open question sent to 9 OSS projects (Steinmacher et al., 2014d); and, (iv) semi-structured interviews with 35 developers from 13 different projects, including newcomers, dropouts, and experienced members. To analyze the data, we used open coding and axial coding procedures of Grounded Theory (Strauss and Corbin, 2007), which are increasingly used to study the human aspects of Software Engineering (Hoda et al., 2010), for example (Smolander et al., 2008; Dagenais et al., 2010; Treude and Storey, 2010; Viana et al., 2012; Pham et al., 2013; Costa Valentim and Conte, 2014; Dittrich, 2014; França et al., 2014a).

We analyzed the data separately and obtained three different models that, afterwards, were merged in a single model. The model was built to organize the identified barriers, mapping and categorizing them as a taxonomy of barriers faced by newcomers to OSS projects. Presented in Section 3.2.4, the resulting model comprises 58 barriers organized in six categories.

The results presented in this chapter were partially published in different venues. The systematic literature review was published in the 10$^{th}$ International Conference on Open Source Software (Steinmacher et al., 2014c), and an extended version was published by the Information and Software Technologies Journal (Steinmacher et al., 2015c). The partial models obtained from each data source were published at International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE) 2014 (Steinmacher et al., 2014d), the Brazilian Symposium on Software Engineering (SBES 2014)(Steinmacher et al., 2014a), and the 18$^{th}$ ACM Conference on Computer Supported Cooperative Work and Social Computing (CSCW2015) (Steinmacher et al., 2015b). The reader may refer to these publication for additional details on each study.

## 3.1 Research Method

Our study aimed to identify and understand the barriers that hinder newcomers from making their first contribution to an OSS project. We chose a qualitative approach because this phenomena occurs in a complex, social environment, in which the context of its occurrence is important (Dittrich et al., 2007).

Figure 3.1 depicts the method followed. The first data set was collected from students who contributed to OSS projects. From their feedback, we noticed that there were recurrently reported barriers. Motivated by the students' reports, we conducted a systematic literature review to identify the available research related to these barriers. From the literature review, we built a model reporting what was already published in OSS domain. To gather more understanding about the phenomena, and to check how the practice was related to the literature, we surveyed a small set of OSS developers.

We analyzed the data from these practitioners, alongside the data from the students, to create a preliminary barriers model. The results of this preliminary model were used as input for an in-depth investigation. This investigation was made by means of semi-structured interviews conducted with both experienced members and newcomers in order to identify and explore the barriers from different perspectives. By analyzing data from different sources and perspectives, we aimed for a broad understanding of the barriers. After conducting the analysis in separate, we analyzed all the data together to create a merged model.

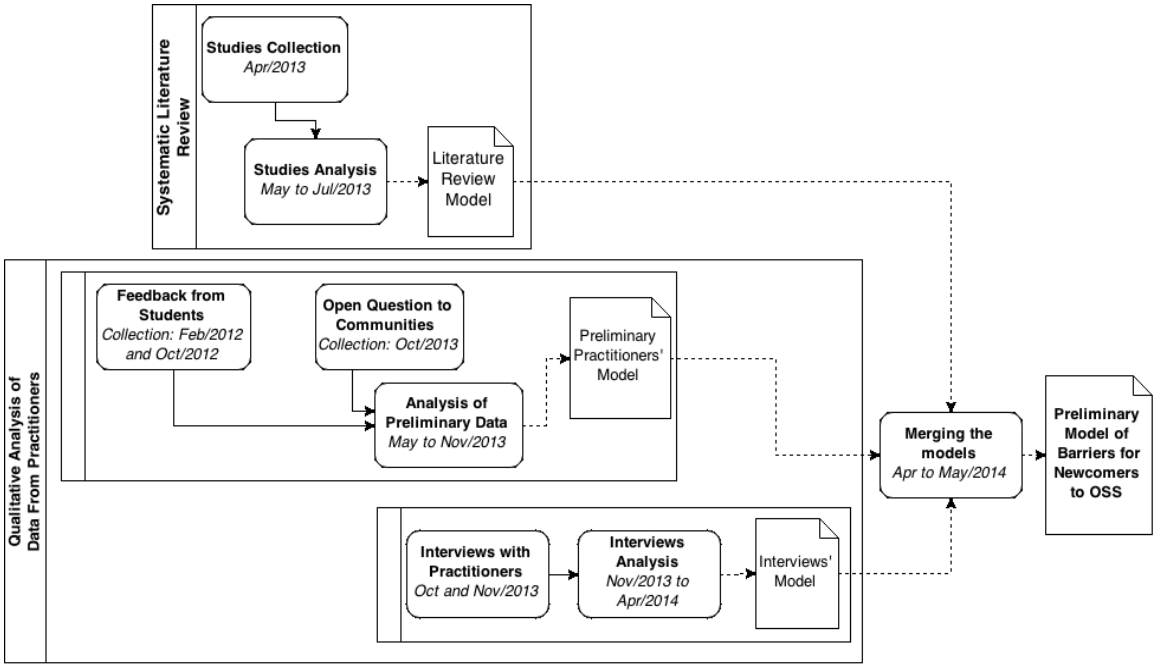



Figure 3.1. Research method followed to identify the barriers faced by OSS newcomers.

During our study, we interacted and received inputs from members and newcomers from 19 different OSS projects. We provide some details about these projects in Table 3.1 (data collected in Feb. 2015, at openhub.net).

Table 3.1. Details of projects that were part of the study.

| Project | Lines of Code (LoC) | Programming language | # of committers** | # of commits** | Project age (years) | Hosted by |
|---|---|---|---|---|---|---|
| **Apache OpenOffice** | 11M | C++ | 39 | 3259 | 14 | Apache |
| **aTunes** | 110K | Java | 1 | 7 | 8 | SourceForge |
| **Audacity** | 273K | C++ | 11 | 419 | 13 | GoogleCode |
| **Cogroo** | 124K | Java | 0 | 0 | 8 | Github |
| **Etherpad** | 42K | JavaScript | 21 | 254 | 3 | Github |
| **GNU Emacs** | 1,5M | Emacs Lisp | 190 | 4152 | 30 | Savannah (GNU) |
| **FreePlane** | 126K | Java | 35 | 514 | 4 | SourceForge |
| **Gephi** | 174K | Java | 4 | 24 | 6 | Github |
| **Integrade** | 190K | Java | 0 | 0 | 11 | Launchpad |
| **JabRef** | 105K | Java | 31 | 326 | 12 | SourceForge |
| **jEdit** | 300K | Java | 9 | 161 | 15 | SourceForge |
| **LibreOffice** | 7M | C++ | 292 | 25311 | 14 | Own host |
| **Moodle** | 2,5M | PHP | 160 | 11450 | 13 | Own host |
| **Mozilla Firefox** | 12M | C++/Javascript/C | 1223 | 59415 | 12 | Own host |
| **Noosfero** | 463K | Ruby/JS | 30 | 1615 | 7 | Gitlab |
| **OpenVPN** | 321K | C | 23 | 318 | 9 | SourceForge |
| **Pardus** | 3,5M | Python | 0 | 8 | 9 | Own host |
| **TextMate** | 94K | C++/Ruby | 17 | 1095 | 6 | Github |
| **zXing** | 51K | Java | 21 | 426 | 7 | GitHub |

** Data from the past 12 months

In the following sections, we detail the methods used to conduct the systematic review and to collect the data from practitioners, followed by details about the data analysis.

### 3.1.1 Systematic Literature Review

We conducted a systematic literature review to identify the empirically evidenced and reported barriers faced by newcomers (Steinmacher et al., 2015c). The goal was to list the barriers that can influence newcomers' joining process. We then aggregated the barriers evidenced by the different studies in a single model. Since we were interested in comparing the empirical data collected from the practitioners, we restricted the systematic review to the OSS domain.

This systematic literature review (SLR) was undertaken based upon guidelines established by Kitchenham and Brereton (2013) and Kitchenham and Charters (2007). The following question guided our SLR:

*What are the barriers that hinder newcomers' contributions in OSS projects?*

After analyzing the terms related to the population and intervention, we derived the keywords and synonyms presented in below to establish the search expression. The synonyms were suggested by experts and extracted from papers we used as a control of this study(Cubranic and Murphy, 2003; Krogh et al., 2003; Ducheneaut, 2005; Jensen et al., 2011; He et al., 2012). Considering the control group previously established, we performed several trials and iterations until coming to a final search expression.

```
("OSS" OR "Open Source" OR "Free Software" OR FLOSS OR FOSS) AND (newcomer OR "joining process" OR newbie OR "new developer" OR "new member" OR "new contributor" OR novice OR beginner OR "potential participant" OR retention OR joiner OR onboarding OR "new committer")
```

We retrieve the studies from the ACM,[16] IEEE,[17] Scopus,[18] and Springer Link[19] digital libraries. These libraries were selected because they index relevant venues for this study, support searches using Boolean expression, and provide access to complete texts. The search was performed in April 2013.

We considered for selection papers that were available for download, written in English, that explicitly mentioned dealing with newcomers to OSS project onboarding, presented empirical results, and were published in peer-reviewed journals or event proceedings. We excluded studies that only analyzed newcomers' motivation for joining a project. Instead, we focused on the issues newcomers face after deciding to contribute.

In addition to querying the digital libraries, we conducted a single step snowballing sampling (Jalali and Wohlin, 2012) following two strategies. The first strategy consisted of 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[20], and personal homepages (when available). The second strategy was backward snowballing sampling. We analyzed the reference lists of all selected papers to find other possible relevant studies. Just one level of snowballing sampling was conducted; that is, we did not apply snowballing sampling for studies found by a previous snowballing process.

After running the query on the digital libraries systems, our resulting sample comprised 291 candidate papers. For each paper, two independent researchers analyzed title, abstract, and keywords. In a consensus meeting, we agreed on 33 papers. We checked other papers published by the authors of these 33 candidate studies, finding 20 other candidate papers. After analyzing these papers, we selected nine relevant papers, resulting in 42 candidate papers. After further analysis, 20 papers were deemed relevant and were considered for data extraction. A summary of the process and the number of papers kept after each step are presented in Figure 3.2.

Given the set of primary studies, we created a list of barriers that each paper evidenced. We related each barrier to information about the study and type of evidence used to verify it. After this identification, each barrier was linked to supporting text segments from the papers in which they were identified. Using the text segments, we classified the barriers applying an approach inspired by open coding and axial coding procedures from Grounded Theory (GT) (Strauss and Corbin, 2007).

Please refer to (Steinmacher et al., 2015c) for additional details about this SLR.

[16] http://dl.acm.org
[17] http://ieeexplore.ieee.org
[18] http://www.scopus.com
[19] http://link.springer.com
[20] http://www.informatik.uni-trier.de/~ley/db/

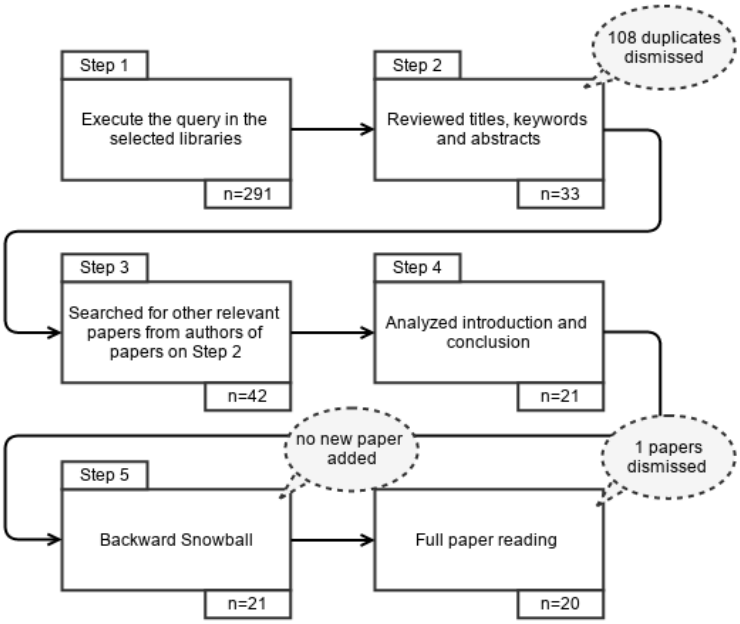



Figure 3.2. Paper selection process.

### 3.1.2 Data from practitioners

This section presents the method used for the qualitative practitioner study. We gathered this data from: (i) students who contributed to OSS projects; (ii) answers to an open question sent to OSS projects; and, (iii) semi-structured interviews with newcomers to and members of OSS projects.

The first source comprised feedback received from four PhD candidates and five undergraduate students who contributed to OSS projects as part of a course assignment. All the student contributors were project newcomers. The PhD candidates were all males, experienced developers, and 30 years old or older. The undergraduate students, including four males and one female, ranged in age between 21 and 24 years old. They were attending the last semester of the Internet Systems course, and therefore were preparing to join the software development industry. The students received an assignment asking them to contribute to an OSS project.

The students contributed to JabRef[21] (2 graduates/1 undergraduate), LibreOffice[22] (3 undergraduates), and Mozilla Firefox[23] (3 graduates) projects. After the conclusion of the assignment, their feedback was collected via an open-ended questionnaire (see Appendix D). The open questionnaire enabled students to debrief and explain the problems they faced while trying to place their first code contribution. The data was

[21] http://jabref.sourceforge.net
[22] http://www.libreoffice.org
[23] http://www.mozilla.org/firefox

collected at the end of the course (February 2012 for the graduate students, and October 2012 for the undergraduate students).

The second data source comprised answers to a questionnaire sent to OSS project developers. The data was obtained from 24 answers to an open question sent to OSS project mailing lists and forums. The messages were posted and the answers received during October 2013. We sent messages to nine projects from different domains: Apache OpenOffice,[24] aTunes,[25] Audacity,[26] LibreOffice, emacs,[27] FreePlane,[28] jEdit,[29] Mozilla Firefox, and OpenVPN[30] (more details about the projects are presented in Table 3.1, Section 3.1).

The questionnaire delivered to the community members comprised two questions designed to profile the contributor (project and contribution time) as well as 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, as detailed in Table 3.2.

Table 3.2. Project to which participants mainly contribute (left) and period of contribution for questionnaire respondents (right).

| Project | Count |
|---|---|
| LibreOffice | 6 |
| OpenOffice | 3 |
| aTunes | 3 |
| Mozilla Firefox | 3 |
| Audacity | 2 |
| jEdit | 1 |
| OpenVPN | 1 |
| FreePlane | 1 |
| Emacs | 1 |
| Did not inform | 3 |

| Time contributing to the project | Count |
|---|---|
| Less than 6 months | 7 |
| Between 6 months and 1 year | 3 |
| Between 1 year and 3 years | 6 |
| More than 3 years | 8 |

The final data collection entailed semi-structured interviews with practitioners. Semi-structured interviews include a mixture of open-ended and specific questions, which are designed to elicit foreseen and unexpected information types (Seaman, 1999). We conducted interviews in order to complement the findings gathered from sources 1 and 2, thereby deepening and broadening our understanding of the newcomers' barriers. We recruited subjects belonging to four different groups:

- *Experienced Members*: project owners, managers, or developers who commit code directly to the software repository;
- *Successful Newcomers*: participants that started to contribute to the project less than one year before the interview;

[24] http://www.openoffice.org
[25] http://www.atunes.org
[26] http://audacity.sourceforge.net
[27] http://www.gnu.org/software/emacs/
[28] http://www.freeplane.org
[29] http://jedit.org
[30] http://openvpn.net

- *Dropout Newcomers*: volunteers that tried to contribute, but gave up;
- *Onboarding Newcomers*: volunteers trying to place their first contribution.

The participants were recruited primarily through mailing list and forums from 15 different projects. In addition, we also directly invited different types of newcomers, identifying them by mining and following projects' mailing lists and issue trackers. Only adults older than 18 were eligible to participate in this study; but we made no distinction related to gender, nationality, or other identity categories. Participants were expected to have software development experience, primarily because we were interested in the barriers to contribute to a project, not to learn how to program.

We interviewed 35 participants from 13 different projects (Pardus,[31] TextMate,[32] zxing,[33] Gephi,[34] jEdit, Moodle,[35] Integrade,[36] Noosfero,[37] Apache OpenOffice, cogroo,[38] etherpad,[39] JabRef, and LibreOffice), including 12 experienced members, 16 newcomers who succeeded, 4 dropout newcomers, and 3 newcomers who were still trying to place their first contributions. More details about the projects are presented in Table 3.1, Section 3.1.

Table 3.3 shows some profiling information of the students and interviewees. The participants received an ID, shown in the first column. The first character of the ID represents the profile of the participant: "S" for student, "E" for experienced member, "N" for successful newcomer, "D" for dropout newcomer, and "O" for onboarding newcomer.

We used a semi-structured format in which a script (interview guide) supported the interviewing process, presented in Appendix B. We started with pilot interviews with five developers involved in Open Source Software Development to adjust the script. Afterwards, we recruited the subjects and conducted the interviews. All interviews were conducted using text-based chat tools, like Google Talk, because the participants used this means of communication in their work, and because it facilitates data collection and interview scheduling.

Each interview was individually conducted and the data was saved on 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. The questions in the interview script guided the interview, but were not necessarily directly asked.

[31] http://www.pardus.org.tr
[32] http://github.com/textmate/textmate
[33] http://github.com/zxing/zxing
[34] http://gephi.github.io
[35] http://moodle.org
[36] http://www.integrade.org.br
[37] http://noosfero.org
[38] http://cogroo.sourceforge.net
[39] http://etherpad.org

Table 3.3. Profile of the participants.

| *ID* | *Hours per week in the project* | *First project?* | *Project* | *Country* | *Years in the project* |
|---|---|---|---|---|---|
| **E1** | < 5 | N | JabRef | France | 8 |
| **E2** | 05-10 | Y | Etherpad | Germany | 3 |
| **E3** | 10-20 | N | JabRef | Germany | 3 |
| **E4** | 05-10 | N | jEdit | Canada | 10 |
| **E5** | 05-10 | N | LibreOffice | Germany | 15 |
| **E6** | > 20 | N | LibreOffice | Hungary | 10 |
| **E7** | > 20 | N | Moodle | Australia | 5 |
| **E8** | > 20 | N | Noosfero | Brazil | 5 |
| **E9** | > 20 | N | Pardus | Turkey | 8 |
| **E10** | 05-10 | N | Cogroo | Brazil | 5 |
| **E11** | < 5 | N | Noosfero | Brazil | 7 |
| **E12** | 05-10 | N | OpenOffice | México | 8 |
| **N1** | < 5 | Y | JabRef | Germany | 0 |
| **N2** | < 5 | Y | Gephi | Brazil | 0 |
| **N3** | 05-10 | Y | Gephi | India | 0 |
| **N4** | 05-10 | Y | Moodle | India | 0 |
| **N5** | < 5 | Y | JabRef | Germany | 0 |
| **N6** | < 5 | Y | jEdit | United States | 0 |
| **N7** | < 5 | Y | TextMate | United States | 0 |
| **N8** | > 20 | Y | Zxing | Greece | 0 |
| **N9** | < 5 | Y | Cogroo | Brazil | 0 |
| **N10** | < 5 | Y | Integrade | Brazil | 0 |
| **N11** | < 5 | Y | Cogroo | Brazil | 0 |
| **N12** | N/I | N | Etherpad | United Kingdom | 0 |
| **N13** | 10-20 | N | LibreOffice | Brazil | 0 |
| **N14** | 05-10 | Y | LibreOffice | Brazil | 0 |
| **N15** | N/I | Y | Etherpad | France | 0 |
| **N16** | 05-10 | N | JabRef | Germany | 0 |
| **D1** | < 5 | N | JabRef | Germany | 0 |
| **D2** | < 5 | Y | OpenOffice | Brazil | 0 |
| **D3** | < 5 | Y | LibreOffice | India | 0 |
| **D4** | < 5 | Y | OpenOffice | India | 0 |
| **O1** | < 5 | N | OpenOffice | India | 0 |
| **O2** | 10-20 | Y | LibreOffice | China | 0 |
| **O3** | < 5 | Y | OpenOffice | Greece | 0 |
| **S1** | N/I | N | Mozilla | Brazil | 0 |
| **S2** | N/I | Y | LibreOffice | Brazil | 0 |
| **S3** | N/I | Y | LibreOffice | Brazil | 0 |
| **S4** | N/I | Y | Firefox | Brazil | 0 |
| **S5** | N/I | Y | Jabref | Brazil | 0 |
| **S6** | N/I | Y | Firefox | Brazil | 0 |
| **S7** | N/I | N | Jabref | Brazil | 0 |
| **S8** | N/I | N | Jabref | Brazil | 0 |
| **S9** | N/I | Y | LibreOffice | Brazil | 0 |

N/I – Not Informed

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, and in some of our published papers (Steinmacher et al., 2014a, 2014d, 2015a, 2015b).

### 3.1.3 Data Analysis

We qualitatively analyzed the data using procedures of Grounded Theory (GT) (Strauss and Corbin, 2007). We applied open coding, whereby concepts are identified and their properties and dimensions are discovered, and axial coding, whereby connections among codes are identified and grouped according to their properties to represent categories.

Although the purpose of the GT method is to construct substantive theories, its use does not necessarily need to remain restricted only to researchers with this goal.

According to Strauss and Corbin (Strauss and Corbin, 2007), a researcher may use only some of its procedures to meet one's research goals.

We split our analysis into two steps. The first (preliminary) step consisted of analyzing the students' feedback and the open questions sent to communities. In the second step, the codes and categories found in the preliminary study were used as seeds for the interview coding. During open coding, we assigned codes to sentences, paragraphs, or revisions. This procedure overlapped the axial coding, in which we identified connections between categories. We executed open and axial coding using the ATLAS.ti[40] tool several times to refine the emerging codes and categories.

In the first step, open coding was conducted in parallel by three researchers. Each researcher independently quoted and coded the documents. After coding, the researchers discussed the quotes and codes until they came to a consensus for the entire document set. This was done to mitigate the bias caused by a single researcher and to reach a common understanding about the code nomenclature and categories. We had two rounds of discussions with three researchers, and two other rounds of discussions involving two researchers. All the decisions finished with a consensus about the codes and categories. The result of this study was a Preliminary Practitioners' Model of the barriers faced by OSS newcomers.

For the second step, we analyzed the data obtained from the interviews. The analysis process 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. This time, a single researcher conducted open and axial coding, and discussed with the other two the questions and proposed new or merged categories.

After obtaining the model from the interview analysis, we iteratively reanalyzed the models obtained from all studies, 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.

## 3.2 Results and discussion

In this section, we report each model separately, and then, the resulting model of OSS newcomer barriers.

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

### 3.2.1 SLR Model

During the SLR, we analyzed 20 studies. From these studies, we identified 16 barriers, grouped into five categories: **finding a way to start; social interactions**; **newcomers' previous knowledge**; **documentation**; and **technical hurdles**. Table 3.4 shows the barriers identified for each category and the studies that evidenced them. The categories are briefly described in the following.

Table 3.4. Studies that evidence each barrier.

| Category | Barrier | Studies |
|---|---|---|
| Finding a Way to Start [5 studies] | Finding an appropriate task/issue to start with [4 studies] | (Krogh et al., 2003; Capiluppi and Michlmayr, 2007; Park and Jensen, 2009; Ben et al., 2013) |
| | Difficulty to find a mentor [1 study] | (Canfora et al., 2012) |
| Social Interaction [12 studies] | Lack of social interaction with project members [7 studies] | (Ducheneaut, 2005; Bird et al., 2007; Bird, 2011; Qureshi and Fang, 2011; Zhou and Mockus, 2011, 2012; He et al., 2012) |
| | Not receiving a (timely) answer [6 studies] | (Krogh et al., 2003; Stol et al., 2010; Jensen et al., 2011; Steinmacher et al., 2012b, 2013b; Zhou and Mockus, 2012) |
| | Receiving an improper answer [3 studies] | (Stol et al., 2010; Jensen et al., 2011; Steinmacher et al., 2013b) |
| Newcomers' Previous Knowledge [8 studies] | Lack of technical experience [6 studies] | (Krogh et al., 2003; Ducheneaut, 2005; Bird et al., 2007; Bird, 2011; Schilling et al., 2012; Singh, 2012; Zhou and Mockus, 2012) |
| | Lack of domain expertise [2 studies] | (Krogh et al., 2003; Stol et al., 2010) |
| | Lack of knowledge on project practices [1 study] | (Schilling et al., 2012) |
| Documentation Problems [4 studies] | Outdated documentation [2 studies] | (Stol et al., 2010; Steinmacher et al., 2012b) |
| | Unclear code comments [1 study] | (Stol et al., 2010) |
| | Information overload [3 studies] | (Cubranic et al., 2005; Park and Jensen, 2009; Stol et al., 2010) |
| | Lack of documentation [1 study] | (Stol et al., 2010) |
| Technical Hurdles [6 studies] | Software architecture complexity [3 studies] | (Cubranic et al., 2005; Park and Jensen, 2009; Stol et al., 2010) |
| | Code complexity [2 studies] | (Bird et al., 2007; Midha et al., 2010) |
| | Issues setting up a local workspace [1 study] | (Stol et al., 2010) |

**Finding a way to start.** Newcomers need support in finding a task and the proper artifacts to change. We found that, from the communities perspective, newcomers should be able to find the most appropriate task themselves (Krogh et al., 2003). However, some studies showed that newcomers need special attention (Capiluppi and Michlmayr, 2007; Ben et al., 2013). This category represents the problems that newcomers face when trying to find the right way to begin contributing. The category includes two barriers: **finding an appropriate task/issue to start with**, and **difficulty to find a mentor**.

**Social interaction**. This category groups the barriers related to the way newcomers interact with the community, including issues related to which members they exchange messages with, the size of their contact network, how they communicate, and how the community members communicate with them. These barriers were mostly evidenced from historical data mined from software repositories. Within this category, we found evidences of three different barriers that can influence newcomers: **lack of social interaction with project members**; **not receiving a (timely) answer**; and **receiving an improper answer**.

**Newcomers' previous knowledge.** This category comprises the barriers related to the experience of the newcomers regarding the project and the way they show this experience when joining a project. It includes domain, process, and technical previous skills. We classified the barriers found into: **lack of technical experience**; **lack of domain expertise**; and **lack of knowledge on project practices**.

**Documentation problems.** This category refers to the need to learn the project's technical and social aspects in order 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 large amount of documentation leads to information overload. Finding outdated documentation or getting lost in a huge amount of information can lead to demotivation. The barriers under this category define which documentation problems have been evidenced as possible barriers to newcomers to OSS projects, and include: **outdated documentation**; **unclear code comments**; **information overload**; and **lack of documentation**.

**Technical hurdles.** This category consists of the project's technical barriers that arise when newcomers are dealing with the code. To contribute, a newcomer usually needs to change existing source code. Therefore, it is necessary for newcomers to have enough knowledge about the code to begin their contributions. This category includes barriers related to **issues setting up a local workspace, software architecture complexity,** and **code complexity.** The main complaint was that code structure was hard to understand, and learning it takes too much time. A study evidenced that newcomers had difficulties setting up their environment (Stol et al., 2010).

The category most thoroughly studied is **social interactions**, accounting for 12 studies, followed by **newcomers' previous knowledge,** with eight studies. The other categories range from four to six related studies each. So far, we can notice that the literature strongly focuses on social issues.

Due to the nature of the approach to establishing the model, there was at least one paper associated to each problem. Considering the most studied barriers, we found that the most evidenced are: (1) **lack of social interaction with project members**, belonging to the **social interaction** category; (2) **lack of technical experience**, belonging to the **newcomers' previous knowledge** category; and, (3) **not receiving a (timely) answer,** belonging to the **social interaction** category. Overall, fewer evidences individually verify technical-related barriers, which include **technical hurdles** and difficulty **finding a way to start**.

### 3.2.2 Preliminary Practitioners' Model

The Preliminary Practitioners' Model resulted from the analysis conducted from Sources 1 (student feedback) and 2 (answers to the to open question). This model comprised seven categories, along with 33 barriers. Table 3.5 presents an overview of

these categories, as well as the number of documents, quotes, and barriers coded. The document count is also reported in terms feedback count 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 3.5. Overview of Categories that Emerged.

| Category | # of documents (feedback/question) | # quotes (feedback/question) | # of barriers |
|---|---|---|---|
| Finding a way to start | 11 (8 / 3) | 22 (18/4) | 3 |
| Social interactions 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 |
| Documentation problems | 15 (8 / 7) | 23 (15/8) | 10 |
| Issues setting up workspace | 8 (4 / 4) | 15 (10/5) | 4 |
| Code issues | 15 (7 / 8) | 21 (11/10) | 5 |

**Finding a way to start.** In this category, the barriers found were the same as those the SLR identified. The only exception was the problem related to **outdated list of bugs**, which was reported during students' feedback sessions. In Table 3.6, we present the evidenced barriers according to the data sources and respondents' profiles. We can see that students who were onboarding to the project largely reported barriers in this category.

Table 3.6. Finding a way to start barriers quotes organized by data source and time in the project

| | Source 1: | Source 2: Open Questions | | |
|---|---|---|---|---|
| Background/ Time in the project | Feedback Students | 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 | • | • | | • |

**Social interactions issues.** In Table 3.7, we can observe that social issues were reported by the students and by community members who recently joined the projects. Two barriers evidenced in the SLR did not appear in these studies. However, the other barriers confirmed the SLR evidence.

Table 3.7. Social interactions issues quotes per data source and time in the project.

| | Source 1: Feedback | Source 2: Open Questions | | |
|---|---|---|---|---|
| Background/ Time in the project | Students | 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 open question answers, we identified issues related to newcomers' behavior that can hinder their onboarding. We identified two barriers under this category, as presented in Table 3.8.

Table 3.8. Newcomers' behavior barriers organized.

| Data Source<br>Background/<br>Time in the project | Source 1:<br>Feedback<br>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' previous knowledge** appeared as a category in this model. Five barriers detailing this category were identified in 12 documents analyzed. Both newcomers and community members recognized previous knowledge as a barrier that hindered newcomers' onboarding, as it can be observed in Table 3.9.

Table 3.9. Newcomers' technical knowledge barriers per data source and time in the project.

| Data Source<br>Background/<br>Time in the project | Source 1:<br>Feedback<br>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 | • | | | |

**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 3.10 reports the barriers and who reported them.

Table 3.10. Documentation problems barriers per data source and time in the project.

| Data Source<br>Background/<br>Time in the project | Source 1:<br>Feedback<br>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 | | | • | |

**Issues setting up the workspace**. To modify the application it is necessary to locally build the application first, which 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 3.11. In the table, it is also possible to observe the data source that evidenced the barriers.

Table 3.11. Issues setting up the workspace barriers per data source and time in the project.

| Data Source<br>Background/<br>Time in the project | Source 1:<br>Feedback<br>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 | • | | | |

**Code issues**. We also identified problems related to code 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 understanding the architecture/code structure**. Table 3.12 presents the barriers split according to data source.

Table 3.12. Code issues reported per data source and time in the project.

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

### 3.2.3 Practitioners' Resulting Model

The result of the analysis was the emergence of 50 barriers grouped into 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 barriers' evidence.

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

Table 3.13. Barriers that emerged from a qualitative analysis of data categorized as **newcomers' orientation**.

| Barriers | Dropout | Newcomers | Experienced |
|---|---|---|---|
| Finding a task to start with | • | • | • |
| Reproducing issues | • | | |
| Finding the correct artifacts to fix an issue | • | • | • |
| Finding a mentor | • | • | • |
| Poor "How to contribute" available | • | • | • |
| Newcomers don't know what is the contribution flow | | • | |

**Newcomers' characteristics.** This category comprises two other subcategories: **newcomers' behavior**, with ten barriers (Table 3.14); and **newcomers' previous knowledge** (Table 3.15), 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 most of them. Regarding **newcomers' previous knowledge**, the interview analysis confirmed the barriers identified in other studies.

Table 3.14. Barriers that emerged from a qualitative analysis of data categorized as newcomers' behavior.

| Barriers | Dropout | Newcomers | Experienced |
|---|---|---|---|
| Lack of proactivity | • | • | • |
| Lack of patience | | • | |
| 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 meaningful/ correct message | | | • |

Table 3.15. Barriers that emerged from a qualitative analysis of data categorized as **newcomers' previous knowledge**.

| Barriers | Dropout | Newcomers | Experienced |
|---|---|---|---|
| Proper knowledge in the programming language | | • | • |
| Knowledge on technologies and tools used | | • | • |
| Knowledge on versioning control system | | • | • |
| Experience on unit testing | | • | |
| Choosing the right development tools | | • | |

**Reception Issues.** OSS communities' receptivity was also evidenced as a barrier, which can 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 even result in newcomers' dropping out. We could identify four barriers, presented in Table 3.16. These barriers can be compared to the barriers identified in the previous studies' social interaction categories.

Table 3.16. Barriers that emerged from a qualitative analysis of data categorized as **reception issues**.

| Barriers | Dropout | Newcomers | Experienced |
|---|---|---|---|
| Receiving answers with too advanced/complex contents | | • | • |
| Delayed answers | | • | • |
| Not receiving an answer | | | • |
| Impolite answers | | • | • |

**Cultural differences.** When OSS development involves global software development, people from different cultures need to collaborate. Yet, these differences can result in interaction problems. In our analysis, three subjects reported that some newcomers faced cultural barriers during onboarding. We could find two barriers under this category, shown in Table 3.17.

Table 3.17. 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 is considered rude | | • | |

**Documentation problems.** We found ten barriers related to documentation problems, as presented in Table 3.18. The barriers identified in the interviews had been already evidenced in other data sources.

Table 3.18. 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 documents | | • | |
| Lack of documentation on setting up workspace | | • | |

**Technical Hurdles.** This category presented the highest number of barriers evidenced during the analysis. We placed all the problems newcomers face when dealing with source code in a single category. To do so, we further classified these barriers into three subcategories: **code/architectural hurdles** (Table 3.19), with seven barriers; **change request hurdles** (

Table 3.20), with four barriers; and **local environment setup hurdles** (

Table 3.21), with four barriers.

Table 3.19. 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/instability | | • | • |
| Codebase size | • | • | • |
| Understanding architecture/code structure | | • | • |
| Understanding the code | • | • | • |
| Understanding flow of information | | • | |

Table 3.20. Barriers that emerged from a qualitative analysis of data categorized as **change request hurdles**.

| 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 3.21. Barriers that emerged from a qualitative analysis of data categorized as **local environment setup hurdles**.

| Barriers | Dropout | Newcomers | Experienced |
|---|---|---|---|
| Building workspace locally | • | • | • |
| Library dependency | • | | |
| Platform dependency | | • | |
| Finding the correct source | | • | |

### 3.2.4 Preliminary barriers model

After obtaining the model from the interview analysis, 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 evidenced barriers. Once again, we merged some barriers and reorganized the categories. The resulting model aggregates all the intermediate models' evidenced barriers. The

model was obtained after the composition of the axial coding analysis, and each leaf code is a concept grounded in the data found during open coding.

The model, presented in Figure 3.3, displays 58 barriers, organized into 6 categories. The numbers that appear after the name of each barrier correspond to the amount of times (different documents) the barrier was 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 (out of 9); number of mentions in the open questions (out of 24); and number of interviewees that reported the barrier (out of 35). In parenthesis, we provided the number of projects in which the barriers were evidenced, considering only the data from practitioners. A major strength of this representation is the combination of literature and new empirical evidence. The barriers which were identified from our analysis, and not found in our SLR are presented in Table 3.22.

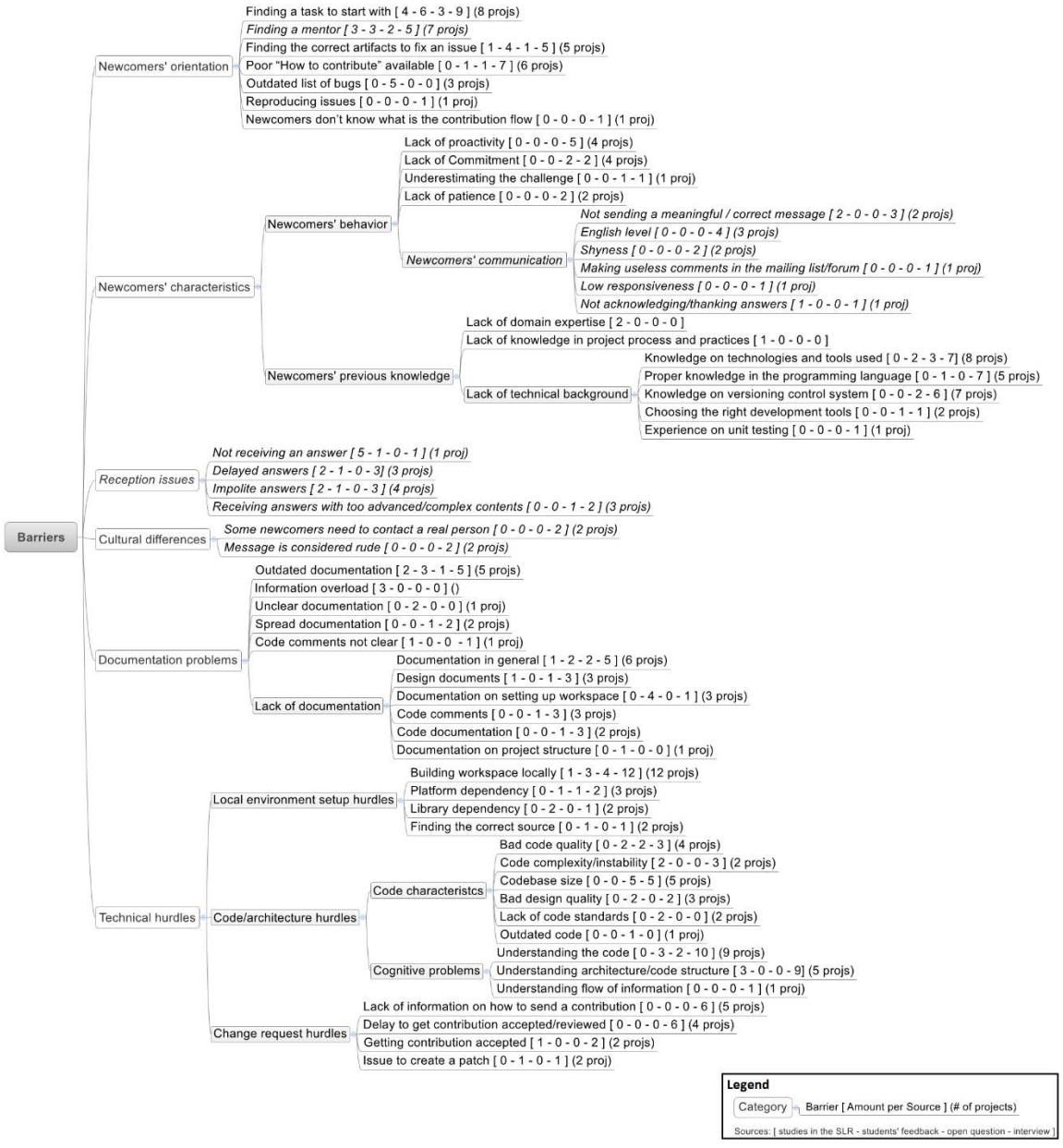



Figure 3.3. Preliminary barriers model for newcomers to OSS.

Table 3.22. Barriers evidenced in our analysis but not found in the Systematic Literature Review

| Barrier | Students' Feedback | Open Question | Interview |
|---|---|---|---|
| **Newcomers' orientation** | | | |
| Poor "How to contribute" available | • | • | • |
| Outdated list of bugs | • | | |
| Reproducing issues | | | • |
| Newcomers don't know what is the contribution | | | • |
| **Newcomers' characteristics** | | | |
| **Newcomers' behavior** | | | |
| Lack of proactivity | | | • |
| Lack of Commitment | | • | • |
| Underestimating the challenge | | • | • |
| Lack of patience | | | • |
| **Newcomers' communication** | | | |
| English level | | | • |
| Shyness | | | • |
| Making useless comments in the mailing list/forum | | | • |
| Low responsiveness | | | • |
| **Newcomers' previous knowledge** | | | |
| **Lack of technical background** | | | |
| Knowledge on technologies and tools used | • | • | • |
| Proper knowledge in the programming language | • | | • |
| Knowledge on versioning control system | | • | • |
| Choosing the right development tools | | • | • |
| Experience on unit testing | | | • |
| **Reception issues** | | | |
| Receiving answers with too advanced/complex contents | | • | • |
| **Cultural differences** | | | |
| Some newcomers need to contact a real person | | | • |
| Message is considered rude | | | • |
| **Documentation problems** | | | |
| Unclear documentation | • | | |
| Spread documentation | | • | • |
| **Lack of documentation** | | | |
| Documentation on setting up workspace | • | | • |
| Code comments | | • | • |
| Code documentation | | • | • |
| Documentation on project structure | • | | |
| **Technical hurdles** | | | |
| **Local environment setup hurdles** | | | |
| Platform dependency | • | • | • |
| Library dependency | • | | • |
| Finding the correct source | • | | • |
| **Code/architecture hurdles** | | | |
| **Code characteristics** | | | |
| Bad code quality | • | • | • |
| Codebase size | | • | • |
| Bad design quality | • | | • |
| Lack of code standards | • | | |
| Outdated code | | • | |
| **Cognitive problems** | | | |
| Understanding the code | • | • | • |
| Understanding flow of information | | | • |
| **Change request hurdles** | | | |
| Lack of information on how to send a contribution | | | • |
| Delay to get contribution accepted/reviewed | | | • |
| Issue to create a patch | • | | |

### 3.2.5 Analysis considering projects' characteristics

To verify to what extent the barriers identified would be found in different projects, we analyzed the findings according to the project that the practitioners belonged to. We grouped the projects according to certain characteristics and observed whether the barriers occurred in projects with their own specific characteristics. We analyzed four

characteristics: programming language, size in line of codes (LoC), age, and forge where the project is hosted. In Table 3.23, we present these characteristics, the way we grouped the projects, and the number of projects per group. The information used to characterize the projects was gathered from OpenHub[41] in Sept. 2014. It is important to note that in our analysis we did not consider categories containing only a single project.

Table 3.23. Projects' characteristics.

(a) Distribution according to programming language

| Programing language | # of projects |
|---|---|
| Java | 9 |
| C++ | 4 |
| PHP, JavaScript Ruby, C, Multiple languages | 1 each |

(b) Distribution according to project size

| Size (LoC) | # of projects |
|---|---|
| More than 1,000 KLoC | 6 |
| From 200 to 500 KLoC | 5 |
| From 100 to 200 KLoC | 4 |
| Less than 100 KLoC | 3 |

(c) Distribution according to project age

| Age | # of projects |
|---|---|
| More than 10 years | 8 |
| From 5 to 10 years | 9 |
| Less than 5 years | 2 |

(d) Distribution according to hosting forge

| Hosted by | # of projects |
|---|---|
| Github[42]/Gitlab[43] | 6 |
| SourceForge | 5 |
| Own hosting | 6 |
| Launchpad[44], Google Code[45] | 1 each |

In the following subsections, we detail the findings according to projects' characteristics. The evidences presented are backed by the data gathered from practitioners. We present a summary of the barriers evidenced per project and according to the project characteristics in Appendix E

#### *3.2.5.1 Reception issues*

With the exception of **not receiving an answer**, all the other barriers under reception category were identified in at least four projects. Only participants related to JabRef reported **not receiving an answer** as a barrier for newcomers. More interestingly, one of the evidences was collected from an interview with a core member, and the other was gathered from a student trying to contribute to the project. In other words, the community is aware of the issue. It is possible that the problem was related to the JabRef community's structure, which is a bazaa with contributions coming from many volunteers, no sponsors, and no paid contributors.

[41] http://www.openhub.net
[42] http://www.github.com
[43] https://about.gitlab.com
[44] https://launchpad.net
[45] https://code.google.com/

When grouping the projects by characteristics, we noticed that reception issues crosscut all kinds of projects. One exception was found with regard to age. There was only 1 citation to reception issues (**impolite answers**) by participants of projects less than 5 years old. However, this can be a sampling issue, as there were just 2 projects with less than 5 years, from which we interviewed 2 experienced members, and received one open question answer.

#### *3.2.5.2 Newcomers' characteristics*

From the **newcomers' characteristics** category, what drew our attention was the **experience on unit testing** barrier, which was mentioned only by a Gephi project newcomer. This is an example of a barrier that is related to a project's specificities. A barrier like this will only appear in projects that use unit tests, like Gephi. Conversely, the other **lack of technical background** barriers – **knowledge on technologies and tools used**, **knowledge on versioning control system**, and **proper knowledge in the programming language** – were mentioned by practitioners of more than 5 projects, thereby crosscutting projects with different characteristics, and were mainly reported in the experienced member interviews.

An interesting finding is that the number of barriers under this **newcomers' behavior** category grew according to the size of the project and age, and appeared mostly on projects hosted on their own forge. These are characteristics of established projects, which possibly require more intensive study and commitment from newcomers. Moreover, the experienced members of such projects possibly had interacted with a larger number of newcomers interested in joining the projects.

#### *3.2.5.3 Newcomers' orientation*

Regarding orientation issues, we found that **finding a task to start with** and **finding a mentor** were repeatedly reported, appearing in more than seven projects each. This category's barriers was identified in different projects, regardless of size, age or programming language used.

The only difference we found in this category concerns the hosting forge. We identified mentions to all the 7 barriers under **newcomers' orientation** in the data obtained from participants whose projects were hosted on their own forges. On the other hand, for those projects hosted by GitHub we saw few mentions of these barriers. This can be attributed to the social coding paradigm introduced by GitHub, which offers a number of social features that make unique information about users and their activities visible within and across open source software projects (Tsay et al., 2014).

#### *3.2.5.4 Documentation problems*

Subjects from almost all projects reported **documentation problems** as barriers for newcomers. They mostly complained about **lack of documentation** and **outdated documentation**. We found a single evidence for **unclear documentation** (LibreOffice) and **documentation on project structure** (Mozilla Firefox), however we could not link these barriers to anything specific in the projects.

By analyzing the projects by their characteristics, it seems that older, larger, and self-hosted projects had more documentation-related barrier reports. It would be interesting to further investigate it. It is understandable that older projects have more documentation, potentially spread across different sites, as well as a higher number of outdated documents. Moreover, there should be a higher demand of documentation for projects with larger codebases, because more effort is necessary to understand its structure. Thus, it is understandable that there are more barriers related to documentation uncovered for older and larger projects. However, since most of these projects have established communities, they should invest in maintaining and creating proper documentation, and organizing it in such a way that benefits newcomers as well as other project contributors.

#### *3.2.5.5 Code/architectural hurdles*

For **code/architectural hurdles**, once again, we could not find any issue that could be related to a specific project. We highlighted **understanding the code** barrier, which was reported by people from eight different projects with different characteristics. In fact, the barriers under **cognitive problems** crosscut all project kinds. This barrier's recurrence was expected (we even expected it to be more acute), since this is a problem inherent to software development (Salviulo and Scanniello, 2014).

We at least expected different sets of barriers when analyzing projects with different codebase size or programming language (*Java vs. C++*). However, we could not find anything that deserved highlight. Surprisingly, the only great difference found was related to project age. The number of barriers reported for projects with more than 10 years (9 barriers) was much higher than from those projects with 6 to 10 years (2 barriers). It possibly relates to Lehman Laws of Software Evolution (Lehman, 1996), which, among other laws, states that code quality declines and code complexity increases as a software project evolves.

#### *3.2.5.6 Local environment setup hurdles*

Within the category **local environment setup hurdles**, we found the barrier that was evidenced in the highest number of projects: **building workspace locally**. We evidenced this barrier in 12 different projects, in data from all the sources, and from all different

profiles. These barriers are agnostic, appearing in projects with different characteristic combinations. This indicates that this barrier is important and present in a large number of projects.

#### *3.2.5.7 Change request hurdles*

We evidenced barriers related to change requests in seven different projects, primarily in the interviews. **Lack of information on how to send a contribution** and **delay to get contribution accepted/reviewed** are the most widely reported barriers within this category, appearing in reports from participants of five and four projects, respectively.

Projects hosted by GitHub/GitLab evidenced only two barriers related to change requests, whereas all four barriers in this category were evidenced in the projects hosted by Source Forge or self-hosted. The "fork & pull request" model adopted by GitHub and GitLab can possibly explain this difference. The fork & pull model lets anyone fork an existing repository and push changes to their personal fork without requiring access be granted to the rights to commit to the repository.

#### *3.2.5.8 Discussion*

We found that some barriers depend on a project's characteristics or singularities. For example, lack of **experience on unit testing** is specific to projects that apply unit testing.

The most distinctive characteristics identified in our data were projects' age and size. Projects with a larger and older codebase seem to present more, and higher, barriers, which can be clearly related to the Lehman Laws (Lehman, 1996). However, it is important to notice that there could be a sampling issue, since most of the participants belong to projects with such characteristics.

Some interesting findings concern project hosting site. We found indications that projects hosted by GitHub present fewer documentation, orientation, and change requests related barriers. We expected that such environments would lower social barriers, but we could not conclude this from our data. As the focus of our study was not to observe such differences, future works may uncover even more differences, especially regarding social coding environments.

### 3.2.6 Better Understanding the Social Barriers

In the model presented in Figure 3.3, we noticed a high number of technical barriers compared to the number of barriers related to the social relationship between newcomers and community members. This imbalance can be explained by the characteristics of OSS communities, which demand contributors and tasks with specific technical skills and

knowledge requirements. We observed that the social barriers identified are fairly similar to those evidenced in other domains. Thus, some of the solutions and mitigation strategies used in those contexts could be adapted to and adopted in OSS communities. In this section, we revisit the social barriers reported in the previous section, linking them to other studies conducted in other CSCW domains. We published the results presented in this section at ACM CSCW 2015 (Steinmacher et al., 2015b).

**Reception issues** were evidenced in all three data sources and reported both by newcomers and experienced members. **Not receiving an answer** was a largely reported barrier. During the feedback sessions, students reported that their forum post was never answered, and thus they worked on an incorrect issue: "*They never answered our forum post. We spent a lot of effort in something that was already being done...*" [S5] In the interviews, an experienced member highlighted: "*In my opinion, the first [barrier] is not getting any reply*" [E1]. We also identified studies in our SLR focusing on the reception barrier (Krogh et al., 2003; Singh, 2012; Steinmacher et al., 2012b, 2013b). Analyzing the Freenet project, Krogh et al. (2003) found that "... *only 29 (10.5%) participants did not receive any reply to their initial posting and subsequently did not appear on the developer list again.*" Singh (2012) reported a similar behavior: "*non returning newcomers can be attributed to not receiving a response...*" CSCW research similarly studies this particular barrier, and the results reported are in accordance with those in the OSS literature and evidenced in our qualitative analysis. Joyce and Kraut (2006) analyzed newcomers' posts to Usenet and found that newcomers who got a reply to their first posts were 12% more likely to post to the community again. Analyzing newcomers to Slashdot, Lampe and Johnston (2005) evidenced similar results.

Compounding the lack of answers, we also found **delayed answers** as a contribution barrier affecting newcomers' motivation: "*[a problem was] a huge delay to receive an answer. It was necessary to send more than one email to receive an answer after a week. Demotivating. I was about to give up*" [S6] It also bothered a newcomer to the zxing project: *"The biggest 'bottleneck' would have probably been the slow pace in communication... if you have a deadline a few days every now and then it can be quite bothersome"* [N8]. However, we could not find any specific evidence of delayed or late reply in the complementary CSCW literature analyzed for this study. Additional study may be necessary to verify whether this is an issue in other domains.

**Impolite answers** also appeared in the students' feedback and the interviews. For example, an experienced member reported: "*...and of course one more thing is the developers' attitude. Some developers may not be suitable for receiving newcomers, they may get angry pretty quickly and kill the interest of the newcomers. Very few of the newcomers know how to behave against this kind of tough developers*" [E9]. CSCW literature, including the studies related to OSS, confirms this barrier. One of the OSS studies (Singh, 2012) reported that non-returning newcomers can be attributed to condescending response. In the context of Wikipedia, Farzan and Kraut (2013) reported experienced editors' hostile behavior caused demotivated newcomers. Some students

complained about reversions and deletions that occurred without proper/polite explanation. Halfaker et al. (2011), Suh et al. (2009) and Zhu et al. (2012) found that negative feedback in Wikipedia decreased motivation. And Suh et al. (2009) reported that the excessive number of newcomer reverts evidenced the Wikipedia community's growing resistance to new content, especially from occasional and new editors.

**Receiving answers with too advanced/complex contents** was another barrier evidenced in our data obtained from practitioners. In some cases, newcomers receive answers that required in-depth knowledge about the project and technologies. For example, a newcomer reported: "*The reason I didn't find the reply helpful is because they talk a little bit out of my understanding of the project*" [O2]. Another newcomer reported a similar problem: "*I found it awesome to get a quick and nice reaction, but the suggestions I could do seemed a bit far fetched to give to a beginner*" [N8]. We could not find any literature reporting or supporting this specific finding.

Looking at newcomers' communication behavior, three experienced members evidenced **not sending a meaningful/correct message** as a barrier. Community members may not answer a message if they do not understand it: "*in general I answer the questions that are well written … some people post things that… I don't know how to answer. So, I wait until someone else makes an attempt and see if the original poster will make a better effort the second time to post something that I can understand*" [E4]. Singh (2012) studied this problem in OSS forums and demonstrated that the community responds better to informative subject lines, comprehensible posts, and correct messages, which is also studied in other CSCW domains. Burke et al. (2007), for example, analyzed Usenet communities, finding that self-introductions can double the odds of receiving answers. Arguello et al. (2006) also analyzed Q&A history and found that on-topic messages and vernacular language use increased reply likelihood. Joyce and Kraut (2006) also analyzed Usenet communities and reported that newcomers were more likely to receive a response if they asked a question or wrote a longer post.

**Shyness** was evidenced as a barrier by an experienced member. He informed that once he gave up contributing because he was too shy to ask the community: "*I was trying to solve a bug… by myself. I was kind of shy to ask for help"* [E11]. Preece (2004) analyzed the MSN bulletin board and found that 28.3% of the lurkers gave shyness as a reason for not posting. A possible approach to deal with this issue would be "breaking the ice." As soon as newcomers subscribe, a member could approach them; automatic greetings could also be used.

**Finding a mentor** was identified as a social barrier for newcomers to OSS. This was evidenced in all the sources we examined, and was reported both by newcomers and experienced members, as presented in Figure 3.3. One experienced member acknowledged that mentorship would be a good way to support newcomers: *"something that I think would help though, is if the more experienced developers had more time to work together with the newcomers through pair programming or even just mentoring. In my experience it's a very fast way to transmit the knowledge"* [E11]. Ease in locating an expert or a

mentor was also evidenced in the systematic review. Cubranic et al. (2005) reported, "*It can be difficult for newcomers to join such groups [OSS projects] because it is hard to obtain effective mentoring.*" There are also several mentorship-related studies in open collaboration communities. For instance, Musicant et al. (2011) qualitatively analyzed data from Wikipedia's program Adopt-a-user[46] and found that several key mentor functions are missing or not consistently fulfilled. Most adopters focus on establishing their legitimacy rather than proactively guiding, protecting, and supporting the long-term growth of adoptees. Choi et al. (2010) analyzed Wikipedia socialization tactics and found that they rarely assign new members a mentor or provide clear guidance about how to behave.

We believe OSS researchers and practitioners can benefit from these results by using them to design newcomer support strategies. And, by including the CSCW and related research literature context, we provide a starting point to do so.

## 3.3 Threats to Validity

Although we analyzed data from a variety of sources and from different projects, we likely did not discover all possible barriers or provide full explanations of the barriers. We are aware that each project has singularities and that the OSS universe is massive, meaning the level of support and the barriers can differ according to the project or the ecosystem. Our strategy to consider different projects and different developers' profiles aimed to alleviate this issue by identifying recurrent mentions of barriers from multiple perspectives.

In our systematic literature review, we considered any study that empirically evidenced barriers faced by newcomers during their joining process. We did not constrain our search to those papers that focused on the newcomers' first contribution, since the papers did not explicitly profile the newcomers analyzed, and there was a high diversity of projects studied. Therefore, there could be studies focusing on different phases of joining process.

We understand that the use of textual chat as the interview means can be considered a threat. The possibility of context change and the execution of parallel activities that distract the interviewees can be a negative aspect of using this mean. The use of Instant Messengers has been discussed in the social sciences (Opdenakker, 2006; Hinchcliffe and Gavin, 2009), and they point out that there is a set of positive effects of using these tools. In our case, we chose to use this means once the participants are used to the environment (they could choose the IM that they were more used to), and electronic means are the default (and preferred) way of communication in OSS projects.

[46] http://en.wikipedia.org/wiki/Wikipedia:Adopt-a-user

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

During interviews, experienced members were asked to answer questions regarding barriers faced by newcomers when they were seeking to place their first contribution; but, due to memory effects, they may have referred to the whole joining process. To avoid this kind of situation, we reinforced the focus of the research and verified some answers.

Finally, since we sent open invitations to a mailing list, sampling bias affects our interviewees and open question respondents: namely, self-selection bias and social desirability bias. However, we tried to counteract this effect by seeking out different sources and analyzing the answers in context to identify specifics.

## 3.4 Final considerations

In this chapter, 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 organizing the barriers that hinder the contributions from newcomers to OSS projects as a taxonomy. The model is composed of 58 barriers grouped in six different categories, expanding and organizing the existing common knowledge about barriers faced by OSS project newcomers.

Notably, 50 out of the resulting model's 58 barriers were identified in the data from practitioner interviews. Furthermore, less than 30% of the barriers (17 barriers) were evidenced by the literature; and, moreover, only six barriers presented in the model were evidenced in all of the analyzed sources.

OSS communities can benefit from these results by using them to provide appropriate newcomer support. We expect to make communities aware of the problems that can hinder the first contributions, offering them an opportunity to think carefully about newcomer reception. By making newcomers aware of the problems they can face, and of the support strategies the project uses for each category (or barrier), communities can manage newcomers' expectations and projects can benefit from more contributions. A smooth first contribution may increase the total number of successful contributions made by single contributors and, hopefully, the number of long-term contributors.

Based on the categorization proposed in the preliminary barriers model, in the next chapter we evaluate the barriers' organization by developing a portal to support newcomers.

# Chapter 4
# FLOSScoach: a Portal to Support Newcomers' First Contribution

To apply and evaluate the organization of the preliminary barriers model, we developed a portal to guide OSS project newcomers' first contributions. In this sense, we followed the categorization presented in the barriers model to organize the portal. In each portal section, we provided information and links to help newcomers overcome the barriers related to a given category. After developing the portal, we evaluated it by conducting a study relying on: qualitative data from diaries (Naur, 1983; Jepsen et al., 1998); a self-efficacy questionnaire (Bandura, 1986; Laitenberger and Dreyer, 1998); and the Technology Acceptance Model (TAM) (Davis, 1989). We found that the portal supported newcomers by making them aware of the process and guiding them through the existing information, thereby increasing their self-confidence. However, when it came to technical barriers, we found that portal-users faced the same technical hurdles as the participants who did not use it. During the development and the evaluation of the portal, we observed the model's practical application, enabling us to create an updated version of the model.

## 4.1 Research Method

In this kind of research, which generated a theoretical model, it is important to verify how the theory influences and reflects the practice. Since we are dealing with a software development topic related to newcomers' onboarding, we decided to develop a newcomers' portal to apply, evaluate, and improve the model.

To guide our research towards this study's objective, we defined four research questions that we aimed to answer:

- Q1. How can the barriers model be used to build a portal that guides OSS project newcomers first contributions?
- Q2. How would newcomers use the portal to overcome the contribution barriers?
- Q3. Does the use of the portal impact newcomers' self-efficacy?
- Q4. What is this portal's perceived usefulness, ease of use, and likely future use?

Based on these questions, we defined the iterative research method depicted by Figure 4.1.

To answer question Q1, we developed and updated a portal to support OSS project newcomers. We followed an approach similar to action-research (Kemmis et al., 2014), in which the model and the portal could evolve after each iteration. In the method's first iteration, we mapped the preliminary model's barrier organization onto sections of an OSS newcomer portal. We then conducted a second research method iteration.

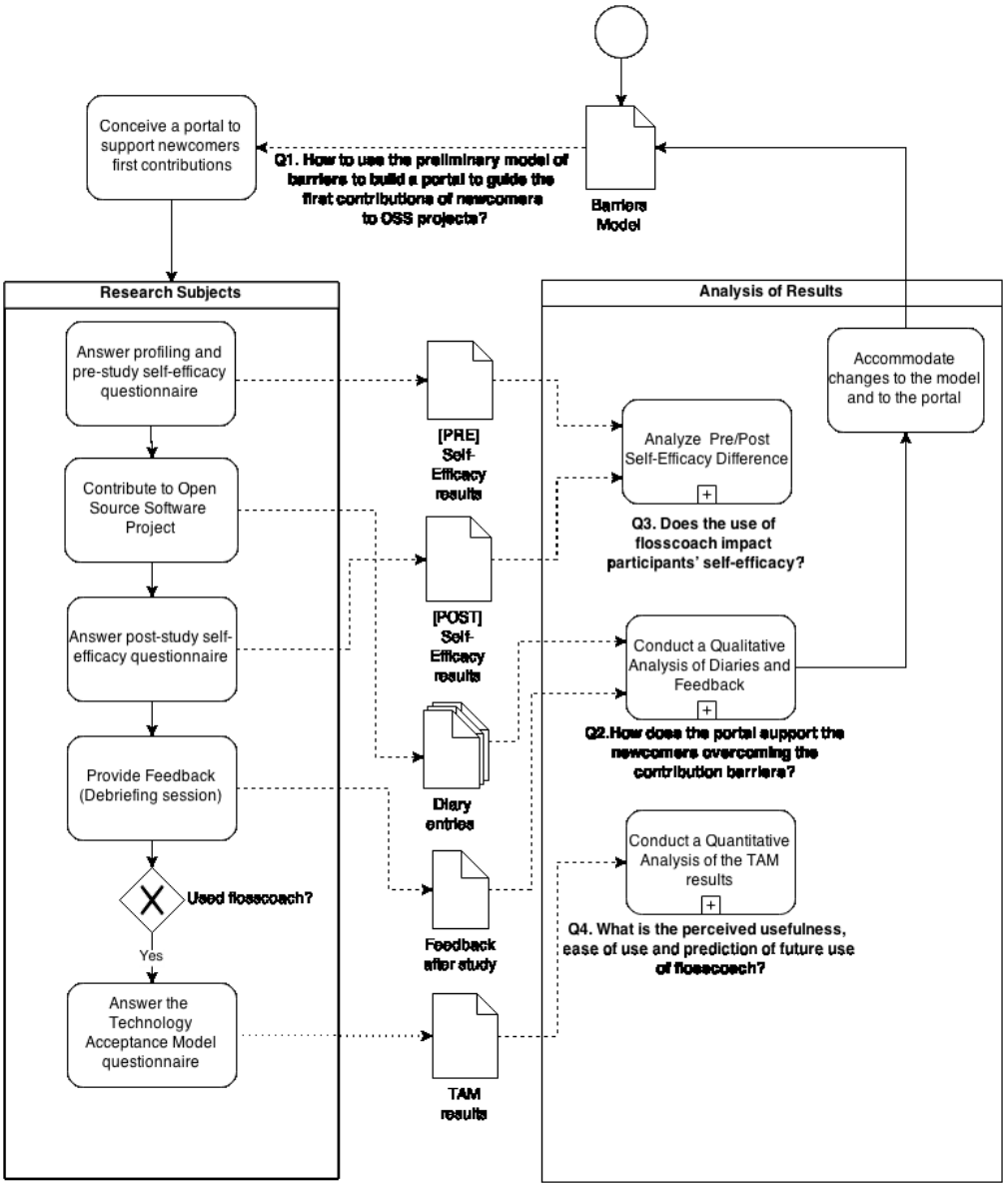



Figure 4.1. Research method for assessing FLOSScoach.

To answer question Q2, we conducted a qualitative analysis using data obtained via diaries and feedback from newcomers trying to place their first OSS project contribution. Details about data collection, analysis, and results are presented in Section 4.2.1.

To answer question Q3, we administered to the participants' pre- and post- study self-efficacy questionnaires, which were quantitatively analyzed. More information about this step and its results are presented in Section 4.3.

Finally, to answer question Q4, we applied the Technology Acceptance Model (TAM) by conducting another questionnaire exclusively with the participants who used FLOSScoach portal. Details about this step and its results are presented in Section 4.4.

As explained beforehand, this study was conducted in two iterations. The first iteration was conducted with a small set of students and projects, and the second iteration was conducted with a larger students and project sample. We detail the subjects and iterations in the following subsection.

### 4.1.1 Subjects and assignment

Since we conducted the study in two different iterations, we present the details individually. Our subjects were undergraduate students. According to a study by Höst et al. (2000), students may provide an adequate model of the professional population and point to a set of benefits that researchers gain from empirical studies. Furthermore, Runeson (2003) identified a similar trend when comparing freshman, graduate, and professional developers.

The students chosen for this study had enough knowledge to fix small bugs in software projects and were motivated to contribute (since their grade depended on it). Furthermore, they joined a real project with real issues, and they interacted with the actual code and community. Moreover, students are potential OSS project contributors, which is why there are currently several programs (for example, GSoC[47], Facebook Open Academy[48]) focusing on attracting them. To guarantee they were newcomers, we verified that the students were not project developers.

For both iterations, we first profiled the users to verify the factors that could influence their contribution and potential challenges. We mainly checked the subjects' experience level, asking them to report their previous industry experience, OSS project experience, and level of knowledge in the project's main programming language. Previous experience in industry can benefit newcomers since they are expected to have some knowledge about the overall development process, technologies, and possible problems they might face. We expect that subjects with prior OSS project experience are aware of the contribution process and common issues. Therefore, people with previous experience would benefit from fewer or lower contribution barriers. We also mapped contributors' gender and age.

[47] https://developers.google.com/open-source/soc/
[48] https://www.facebook.com/OpenAcademyProgram

#### *4.1.1.1 Iteration 1. Students from UTFPR*

Students attending a Software Engineering course at Federal University of Technology – Paraná (UTFPR) received a graded assignment. The students belong to two different majors: five from the 6th semester of Internet Systems Technology, and nine from the 5th semester of Computer Sciences. All participants were newcomers to software development in general. Only five of them worked in the industry, but for less than a year. Only one participant presented previous experience in OSS project contributing. The summary of their profiles appears in Table 4.1.

Table 4.1. Profile of subjects from Iteration 1.

| OSS Project | Case/ Control | Participant | Age | Gender | Industry experience | OSS experience | Programming language** |
|---|---|---|---|---|---|---|---|
| JabRef | Case | FC1_UTF | 20 | M | < 1 year | No | 3 (Java) |
| | | FC2_UTF | 22 | M | < 1 year | No | 3 (Java) |
| | | FC3_UTF | 25 | M | < 1 year | No | 2 (Java) |
| | Control | C1_UTF | 27 | M | None | No | 3 (Java) |
| | | C2_UTF | 22 | M | < 1 year | No | 2 (Java) |
| LibreOffice | Case | FC4_UTF | 21 | F | None | No | 2 (C++) |
| | | FC5_UTF | 21 | M | None | No | 3 (C++) |
| | | FC6_UTF# | 19 | M | None | 1 contribution 1 project | 3 (C++) |
| | | FC7_UTF | 19 | M | None | No | 2 (C++) |
| | Control | C3_UTF | 20 | M | None | No | 3 (C++) |
| | | C4_UTF | 20 | M | < 1 year | No | 3 (C++) |
| | | C5_UTF | 21 | M | None | No | 2 (C++) |
| | | C6_UTF | 20 | F | None | No | 2(C++) |
| | | C7_UTF | 19 | F | None | No | 3 (C++) |

**Note:** Experience with the programming language given in a scale from 0 (no experience) to 4 (expert)

One participant assigned to use the portal (identified by #) decided not to use it. He reported that he accessed the portal just once and decided to use the project page instead. During the debrief session, he reported that he had already contributed to an OSS project before, and found out that all the information was relying on the LibreOffice page.

We directed the participants to specific OSS projects. The chosen projects for the first iteration were LibreOffice[49] and JabRef[50]. These two projects were part of our previous studies, presenting a high number of participants and communities receptive to our research. Moreover, this study's author is a JabRef participant. These two projects are very different in terms of governance, size, and support. LibreOffice is a large, well-known project that is supported by The Document Foundation[51] and some companies. It has an established and responsive community and many supporting strategies in place. The second is an established project with few contributors and a small community, having no foundation or company support and little support for newcomers.

Since the two majors focus on different technologies and languages, we split the participants and directed them to two different projects. For the students majoring in Internet Systems, whose background is in Java technology, we requested they contribute

[49] http://www.libreoffice.org
[50] http://jabref.sourceforge.net
[51] http://www.documentfoundation.org/

code to JabRef. For the students in the Computer Science major, whose background is in C/C++, we requested they contribute code to LibreOffice. We randomly assigned the students into two groups:

- **Case (group A)**: received access credentials to the portal FLOSScoach, receiving FLOSScoach.com page as their starting point;
- **Control (group B)**: did not receive credentials to FLOSScoach, and received the project's webpage as their starting point.

The students received the following assignment, in Brazilian Portuguese:

> You are asked to significantly contribute to *<Name_of_the_project>* project. Your contribution(s) must be fixing bug(s) or implementing new feature(s) according to what is already reported in project's issues tracker. A contribution is considered complete once the code is accepted by the project members and included in the main trunk of the project. This happens after submitting the piece of code implemented, and it passes the review.
> *GROUP A:* [You can find the information you need in the newcomers coach portal, which is www.FLOSScoach.com. You can use any information, tool or mean available in the tool and pointed by the project website, as well as search engines to achieve your goal. Please use the following information to login:]
> *GROUP B:* [All the information you must get can be found in the project web site, which is *<project_homepage>*. You can use any information, tool and mean available and pointed by the project website, as well as search engines to achieve your goal. ]
> You need to share every detail of your journey in your diary. EVERY DETAIL: what have you done, what went well, what went wrong, where did you find etc. All the steps you performed. Do not leave it for the next day; write your diary in parallel with your activities.
> Please create the diary as a shared document in Google Docs. Share the document with igorsteinmacher@gmail.com and, giving write access. Each entry must be identified by the day it was made. The diary will be read often, and comments can be placed requesting more information about the postings made. Responses to comments can be written directly in the text of the diary, without the need for answer as a comment.

It is important to notice that this assignment was part of the course program, and had been applied to evaluate the students in at least four previous editions of the same course. Thus, there was no change on the way the students were graded, or on the scope of the courses in which the assignment was applied. Moreover, the students were not evaluated for their actual contribution, but for the process of contributing. We emphasized it to all the students before the assignment (as it had been done in the previous editions of the course).

The students received the assignment on July 25, 2014, and had one month to complete it. It was part of their assignment to write diaries logging their activities, issues, and everything that they did while working on the assignment. Before receiving the assignment, the students received two trainings:

- Open Source Software Basics: history, what is open source,, community structure;
- Writing Diaries: how to write, what to write, when to write. Includes a practice.

During the study, the course lecturer supported the students solely in terms of the assignment's procedures and methods; he did not provide any technical support or contribution process help.

#### *4.1.1.2 Iteration 2. Students from USP*

The second iteration was conducted during a Software Engineering course of a Computer Science major at University of São Paulo (USP). The initial number of participants was 51 students. The profile was a little different from those participants of Iteration 1, as shown in Table 4.2. The main difference is that the participants had slightly more industry experience.

Table 4.2. Profile of subjects from Iteration 2.

| Project | Case/Control | Participant | Age | Gender | Industry Experience | OSS Experience | Programming Language |
|---|---|---|---|---|---|---|---|
| Amarok | Case | FC01_USP | 28 | M | 0 | No | 3 (C) |
| | Case | FC02_USP | 21 | M | 3 - 4 years | No | 3 (C) |
| | Case | FC03_USP | 21 | M | < 1 year | No | 4 (C) |
| | Case | FC04_USP | 21 | M | 0 | No | 4 (C) |
| | Control | C01_USP | 20 | M | < 1 year | No | 3 (C) |
| | Control | C02_USP | 21 | M | < 1 year | No | 3 (C) |
| | Control | Dismissed (2) | 24 | M | < 1 year | No | 4 (C) |
| | Control | Dismissed (2) | 37 | M | 0 | No | 2 (C) |
| | Control | Dismissed (1) | 23 | M | 1 - 2 years | No | 3 (C) |
| Audacity | Case | FC05_USP | 21 | M | 0 | No | 2 (C++) |
| | Case | FC06_USP | 23 | M | 1 - 2 years | No | 2 (C++) |
| | Case[#] | C04_USP | 26 | M | 0 | No | 2 (C++) |
| | Control | C03_USP | 20 | M | 0 | No | 4 (C++) |
| | Control | C05_USP | 29 | M | 4 - 5 years | No | 3 (C++) |
| | Control | Dismissed (4) | 20 | M | 0 | No | 4 (C++) |
| Empathy | Case | FC07_USP | 29 | M | 0 | No | 2 (C) |
| | Case | FC08_USP | 22 | M | 0 | No | 3 (C) |
| | Case | FC09_USP | 37 | M | > 5 years | No | 3 (C) |
| | Case | Dismissed (3) | 25 | M | 1 - 2 years | No | 2 (C) |
| | Control | C06_USP | 30 | M | < 1 year | No | 3 (C) |
| | Control | Discarded (1) | 23 | M | 2 - 3 years | 1 project 1 year | 2 (C) |
| | Control | Dismissed (2) | 20 | M | 0 | No | 2 (C) |
| | Control | Dismissed (2) | 23 | M | < 1 year | No | 3 (C) |
| | Control | Dismissed (3) | 19 | F | 0 | No | 2 (C) |
| Jabref | Case | FC10_USP | 27 | M | 4 - 5 years | 1 project 1 new feature | 4 (Java) |
| | Case | FC11_USP | 20 | M | 0 | N | 2 (Java) |
| | Case | FC12_USP | 19 | F | 0 | N | 2 (Java) |
| | Case | FC13_USP | 22 | M | 3 - 4 years | N | 2 (Java) |
| | Case | FC14_USP | 20 | M | 0 | N | 2 (Java) |
| | Control | C07_USP | 22 | M | 3 - 4 years | N | 3 (Java) |
| | Control | C08_USP | 20 | F | 0 | N | 2 (Java) |
| | Control | Dismissed (1) | 22 | M | 0 | N | 2 (Java) |
| | Control | Dismissed (2) | 22 | F | < 1 year | N | 3 (Java) |
| | Control | Dismissed (3) | 24 | M | 1 - 2 years | N | 3 (Java) |
| | Control | Dismissed (1) | 21 | M | 2 - 3 years | Many projects 1 year | 2 (Java) |
| LibreOffice | Case | FC15_USP | 30 | M | 1 - 2 years | No | 3 (C++) |
| | Case | Dismissed (1) | 25 | M | 2 - 3 years | No | 2 (C++) |
| | Case | Dismissed (4) | 22 | M | 0 | 1 project 2 years | 2 (C++) |
| | Case[#] | C11_USP | 21 | M | 0 | No | 3 (C++) |
| | Control | C09_USP | 21 | M | 2 - 3 years | Many projects 2 years | 3 (C++) |
| | Control | C10_USP | 20 | M | 0 | No | 2 (C++) |
| | Control | Dismissed (2) | 22 | M | 0 | No | 2 (C++) |
| Vim | Case | FC16_USP | 24 | M | 0 | No | 4 (C) |
| | Case | FC17_USP | 20 | M | 0 | No | 4 (C) |
| | Case | FC18_USP | 19 | M | 0 | No | 2 (C) |
| | Case | Dismissed (1) | 21 | M | 3 - 4 years | No | 4 (C) |
| | Control | C12_USP | 22 | M | 1 - 2 years | No | 3 (C) |
| | Control | C13_USP | 20 | F | 0 | No | 3 (C) |
| | Control | C14_USP | 25 | F | 0 | No | 3 (C) |
| | Control | Dismissed (1) | 25 | M | 0 | No | 2 (C) |
| | Control | Dismissed (1) | 26 | M | 0 | No | 2 (C) |

As one of our goals was to assess Iteration 1 findings, and as we were counting on a higher number of newcomers, we prepared the portal with information from four other OSS projects: Amarok[52], Empathy,[53] Vim,[54] and Audacity.[55] We chose projects used by general audience and written in C/C++.

Once again, considering the programming language background of the participants and their previous experience, we split them first by project and, then, into two groups:

- **Case (group A)**: received access credentials to the portal FLOSScoach, receiving FLOSScoach.com page as their starting point;
- **Control (group B)**: did not receive credentials to FLOSScoach, and received the project's webpage as their starting point.

As shown in Table 4.2, we had an initial 51 subjects. However, we considered only 32 to analyze the results. We dismissed subjects if: (1) no diaries were written; (2) they wrote less than three diary entries; (3) they did not fill out the post-study questionnaire; (4) they did not contribute with code. Two participants assigned to use the portal (identified by #) reported that they did not use the portal, so we redirected them to the control group.

The text of the assignment received was the same as in Iteration 1. In this case, it was the first time that the assignment was applied to the course. In previous editions of the course, the students received a similar assignment, however they were asked to develop new features to non-OSS projects. The evaluation was also related to the process and not to the actual contribution.

The students received the assignment on September 24, 2014, and had one month to complete it. In contrast to the first iteration, Iteration 2 students did not receive any training related to Open Source or writing diaries.

[52] https://amarok.kde.org
[53] https://wiki.gnome.org/Empathy
[54] http://www.vim.org
[55] http://audacity.sourceforge.net

## 4.2 How to use the preliminary barriers model to build a portal to guide the first contributions of newcomers to OSS projects?

We developed the FLOSScoach[56] portal based on the barriers model presented in Chapter 3, and on information collected from project members. Our goal was to organize the information reflecting the categorization of the barriers model.

Initially, we chose two projects to gather information about already-in-place strategies and information aimed to support newcomers overcoming the identified barriers. We chose LibreOffice and JabRef because they presented the highest number of project subjects that participated of the interviews, and because of the facility in accessing project members to gather more information.

To collect details about the strategies and information the projects provided, we first analyzed the existing data collected during the interviews. We had included questions in the experienced members' interview script related to strategies used by the projects, meaning information that would be useful for newcomers as well as suggested contribution process (steps to be followed by newcomers). As a result, we gathered a set of information and strategies that newcomers could use to overcome or lower specific projects' barriers. In addition, we mapped a contribution flow by merging the different projects' suggested steps that newcomers should follow, as reported by interviewees. The resultant flow comprises coarse-grained activities that represent the main steps identified in the data.

Furthermore, to gather more information about JabRef and LibreOffice, we talked to one member of each project. We presented the barriers model to them and asked them to point to information that the project had in place to support newcomers overcoming each barrier. In addition, we conducted a manual inspection in the projects' web pages to find other possible information.

We found a few other resources, which we sent to the developers via email asking them to confirm that those resources would be useful for newcomers. The output was a list of information mapped into the related barrier categories that the support might help newcomers overcome. The information obtained included links to project pages, ways to access the communication channels, documents generated by the project, videos, list of skills needed, and a suggested contribution process. In Table 4.3 we present information or strategy kinds identified for each barrier category.

Our first tentative step was to organize the information gathered by splitting the portal into sections and subsections, in accordance with Table 4.3. However, while

[56] http://www.flosscoach.com

analyzing the structure of the portal, we found that information related to communication was spread across **reception issues**, **newcomers' behavior** and **cultural differences** categories. Therefore, we decided to merge these solutions into one single category that provided information regarding communication. As a result, we created a portal presenting the information in five sections for each project.

Table 4.3. Information and strategies identified per category of barriers.

| Category | Subcategory | Information/Strategy |
|---|---|---|
| Newcomers' orientation | | • Present a suggested contribution flow (step-by-step)<br>• Tasks identified as easy for newcomers<br>• Foster communication via mailing list/Internet Relay Chat (IRC)<br>• Help finding mentor volunteers (some projects have their mentors listed in OpenHatch[57]) |
| Newcomers' characteristics | Newcomers' previous knowledge | • Make it clear what kind of knowledge is needed to contribute project<br>• Add links to tutorials/training in technologies |
| | Newcomers' behavior | • Newcomers need to present themselves and be polite, proactive and objective when communicating<br>• Newcomers must tell how much they know about a problem<br>• List of soft-skills that are expected from newcomers<br>• When you have a problem, follow the flow: search project page and archives → IRC → mailing list |
| Cultural differences | | • People need to adapt themselves to the context<br>• Regional mailing lists (LibreOffice provides) |
| Reception issues | | • Sending polite and meaningful messages should reduce this barrier |
| Documentation problems | | • Organize the links to the documentation<br>• Links to documentation (e.g., documents in project wiki, doxygen[58] generated documentation, Javadoc, opengrok)<br>• Diagrams and figures depicting the code structure and flow<br>• Links to tutorials for workspace setup<br>• Search for solution in the mailing list archives |
| Technical hurdles | Local environment setup hurdles | • Links to tutorials for workspace setup (info)<br>• Search for solution in the mailing list archives |
| | Code/architecture hurdles | • Links to documentation (e.g., documents in project wiki, doxygen generated documentation, Javadoc, opengrok[59])<br>• Link to the project codebase navigator (github/git web/opengrok)<br>• Links to presentations about the project<br>• Links to code conventions and style |
| | Change request hurdles | • Links to submission guidelines/tutorial to submit pull request<br>• Links to information about code licensing |

In Figure 4.2 we show how the barriers model was mapped into the portal sections. This mapping is also presented in the following items:

- **Newcomers' orientation** → How to start
- **Newcomers' characteristics** → Newcomers characteristics
- **Reception issues** and **newcomers' characteristics/newcomers' behavior** (some information) and **cultural differences** → Communication
- **Documentation problems** → Documentation
- **Technical hurdles/local environment setup hurdles** → Technical issues/workspace setup
- **Technical hurdles/code/architecture hurdles** → Technical issues/Deal with code
- **Technical hurdles/change request hurdles** → Technical issues/Submit your changes

[57] http://www.openhatch.org
[58] http://www.doxygen.org/
[59] http://opengrok.github.io/OpenGrok/

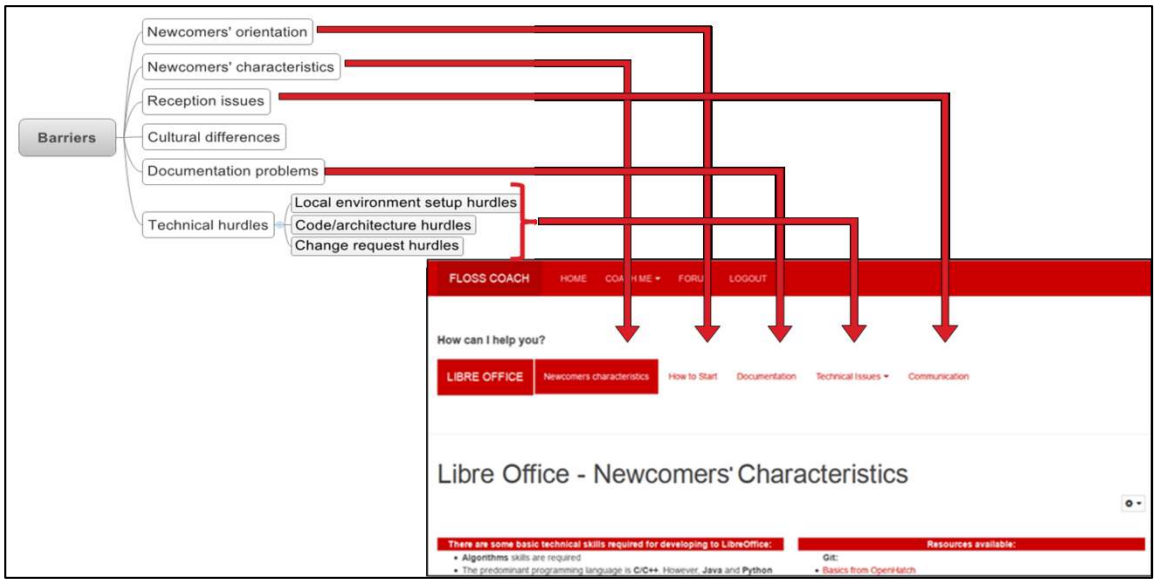



Figure 4.2. Barriers categories mapped to the FLOSScoach sections.

In Figure 4.3 we show the portal accommodating the LibreOffice information. To illustrate another section of the tool, in Figure 4.4 we present the "How to Start" section, highlighting the contribution flow created to guide the newcomers. The newcomer can use the flow to access the other sections of the tool once the box that represents the steps is clickable.

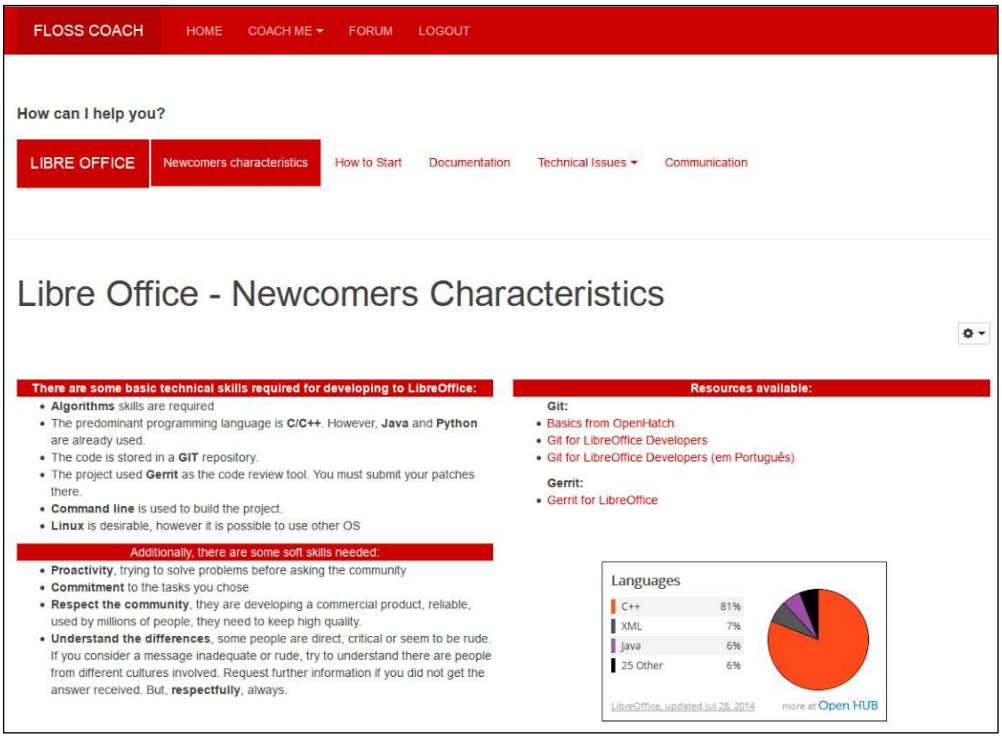



Figure 4.3. FLOSScoach page with information about newcomers' characteristics for LibreOffice.

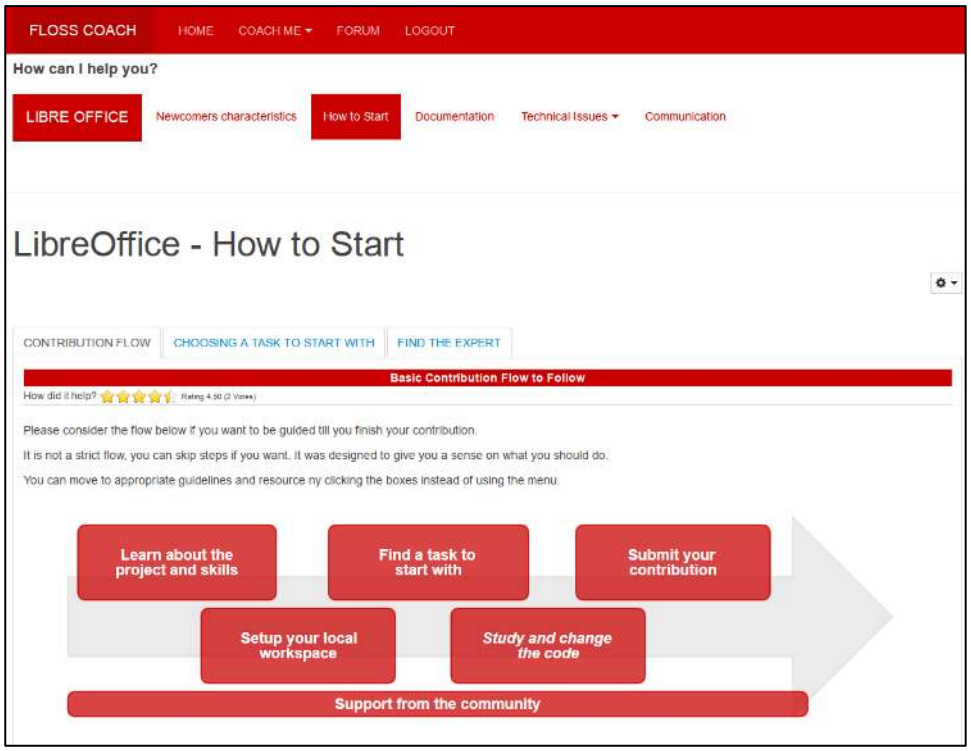



Figure 4.4. FLOSScoach "How to Start" page presenting the suggested contribution flow.

After developing this portal, we started our experimental study with students using FLOSScoach. In the following sections, we explore each of the study's research questions.

### 4.2.1 How does the portal support the newcomers overcoming the contribution barriers?

In this section, we present the data collection details used for answering the research question Q2, as well as the results obtained for each iteration. We analyzed the entries by qualitatively analyzing the data gathered, using open coding and axial coding procedures. It is important to notice that, since all the diaries and feedback sessions were conducted in the students' native language (Brazilian Portuguese), the quotes presented to ground our findings are free translations from the excerpts to English.

### 4.2.2 Gathering qualitative data using diaries

The diary study allows access to everyday behavior in a relatively unobtrusive manner, which affords access to the experience's immediacy, and also provides accounts of phenomena over time (Symon, 2004). Diary studies methodology has been used to collect longitudinal data in many disciplines, including psychology, health, education, anthropology, architecture, etc. (Davidson et al., 2014). There have also been daily diary studies in various settings within the Computer Sciences (Naur, 1983; Adler et al., 1998;

Czerwinski et al., 2004; Kersten and Murphy, 2005; Begel and Simon, 2008; Hess and Wulf, 2009; Davidson et al., 2014). For instance, Begel and Simon (2008) collected data about important daily events from novice Microsoft developers' video diaries.

Diary study participants are asked to keep record about a particular activity. These diaries can be highly structured, with a specific pre-defined task, or unstructured. In unstructured diaries, participants might provide open-ended, stream-of-consciousness narratives about their activities and experiences, which can result in very rich and detailed accounts (Palen and Salzman, 2002).

We chose to conduct a diary study because, as reported by Davidson et al. (2014), we could not observe our participants. Our participants were able to work whenever they desired, conducting the work at a chosen time and place. Therefore, it was not possible to be with them every time they were working on their assignment. Since our goal was to explore how the portal supported newcomers throughout the contribution process, our diary study was unstructured, which enabled participants to write anything they wanted about their journey.

Regular interaction between the investigator and participants is very important, because it helps the subjects to understand the diary entries' importance and researchers' desired level of details. Therefore, we used shared documents to keep the diaries. Each participant created his/her own document, which they shared with the researchers. We constantly followed the entries and provided prompt feedback to the subjects via annotated comments. We guided them, asking for more details about what they posted, asking, for example, how they achieved something, why they made a given decision, or where they found a piece of information.

#### *4.2.2.1 Post-study debriefing session*

We conducted a quick post-study debriefing session with the participants, aiming to complement the information provided by the diaries and clarify possible questions. The method used for collecting such data was different for each iteration.

For Iteration 1, we conducted debrief sessions by means of semi-structured interviews, following the script presented below:

- What were your overall impressions of the experience, considering the positive and negative points of the contribution process?
    - How do you feel about the outcome of this activity?
- What were the main barriers you faced during the process?
    - Detail the issues
- How was your contact to the community?
- What would you suggest to the community to improve the newcomers' experience? (strategies, information, tools, etc.)
- How would you describe the role of FLOSScoach during this process?
    - What are the main benefits and drawbacks of the portal?
    - What are the weaknesses or elements that can be improved in the portal?

We conducted the interviews one day after the students' deadline. The interviews duration ranged from 9 to 16 minutes, with an average of 12 minutes. Afterwards, we transcribed the interviews in order to conduct the qualitative analysis.

In Iteration 2, we conducted an online open questionnaire on the day the participants finished the assignment. We used the same questions as in the Iteration 1 script above.

### 4.2.3 Results for Iteration 1

To answer question Q2, we analyzed the diaries and the debrief session transcripts both from newcomers who used, and newcomers who did not use, the portal. In the following subsections, we report the findings related to question Q2 in more detail, aiming to identify the portal's influence on the newcomers' contribution process. We report the results, categorizing the discussion across three axes: process, social, and technical.

#### *4.2.3.1 Process: the portal played an important role*

The initial feeling of the participants who did not have access to the portal was uncertainty and doubt on how to proceed. They received an assignment, and the big question came to mind: what to do now? Three out of seven participants in control group reported it in the first diary entry, while two other reported this feeling later in the process as well, just after finishing the local workspace setup.

> *"I opened my browser and typed the website address: http://www.libreoffice.org/. I will need to contribute to LibreOffice but I don't have any clue on how to do it"* – C06_UTF (1st sentence of the diary)

> *"... I am a little lost, so I will try a bug that I think I can work with..."* – C02_UTF (in his 1st entry)

> *"It seems that there is no place centralizing all the commands or information, to enable the developers to integrate quickly"* – C03_UTF

> *"I don't know what I was supposed to do after finishing the compilation process. I will watch the video tutorial once again to find it out. I need to define my next steps, I don't know what these steps are"* – C06_UTF

This issue relates to at least two other barriers that our model evidenced: **spread documentation** and **information overload**. These problems can lead newcomers to feel lost when trying to better understand the project, as reported by these two participants on the control group.

*"I am feeling the necessity of finding something that will be my guide during this process, because until this moment I had to search different solutions and information in different places"* – C04_UTF

*"It seems that there are many ways to gather information about the project. I am searching more and more pages and losing the focus of what I was looking for: how to download the code and develop for the project"* – C03_UTF

*"I decided to organize and search for information once again. I noticed that I had too many pages opened and some of them were obsolete."* – C03_UTF

In one case, a student took a completely wrong path, analyzing a different codebase by cloning the wrong repository and trying to set it up. He spent around one third of his available time on this wrong path, but was put back on track when a colleague pointed out his mistake.

The guidance provided by the portal FLOSScoach avoided this kind of situation. At the very beginning, the newcomers who used the portal felt more comfortable and were aware of the expected steps. Four out of six participants (66.67%) of case group reported these feelings at the beginning of their first entry in the diaries, for example:

*"The tool seems to be good, because it solves doubts that range from the skills needed to start to pointing how to submit a contribution."* – FC07_UTF

*"I could check what newcomers need to know regarding the development environment, accessing the links to documentation and relevant guidelines, understanding how to search for help and who to talk to in case of problems and, mainly, accessing the newcomer guide showing the flow and offering support to each step of the process."* – FC04_UTF

*"...the tool helped me a lot, because it gave me an outstanding guidance about what I needed to do and, consequently, made me spend less time and made me more confident"* – FC05_UTF

In addition, we noticed that no participant who used the tool in the contribution process reported barriers related to **newcomers' orientation**, what are the next steps, or feeling lost. During the debrief session, we gathered more evidences of the portal's positive aspects. The participants reported on the portal's organization, and highlighted the role of the "contribution flow" in overcoming the barrier of not knowing "how to start." For example:

*"I accessed FLOSScoach, read the 'How to start', and followed the steps of the suggested flow. Then, I accessed the 'setup your workspace'..."* – FC02_UTF

*"The flow [of the FLOSScoach] was great. I always used it, and from here I accessed the other information. It is easy"* – FC02_UTF

*"That timeline [the flow] is very good. I really liked it. I think that for those who are contributing the first time it is very good, because the person thinks 'what should I do now?' and the answer is there"* – FC07_UTF

*"... the 'How to start', the information about workspace setup and the contribution flow were really helpful."* – FC05_UTF

Something that deserves highlight in the newcomer's first contact with the project is that they can check which **newcomers' characteristics** the project requires, including **technical background** and **newcomers' behavior**. Anticipating what specific technologies they need to know in order to contribute sets the newcomers' expectations, informing them what they should learn to achieve their goals. In our study, this was clearly evidenced by four participants (66.67%) using FLOSScoach, for example:

*"checking the skills that newcomers needed to know about the development, either technical – like language, versioning control system and preferable operating system – and behavior – respect, commitment and proactivity... I don't know the code review tool, Gerrit, so I will need to know more about it..."* – FC04_UTF

*"Among the skills expected to contribute to LibreOffice, it is said that the predominant language is C/C++ and that Java and Python are also used... other thing that called my attention was that Gerrit is used to review the code, and I never used such tool."* – FC07_UTF

*"I clicked 'check your skills' and the skills needed to contribute to the project were shown. Among them there was the languages C, C++, Java and Python, and the tool Git, that I never used before, and needed to learn how to use..."* – FC05_UTF

Making the newcomers aware of the technologies/languages used was well received, and created the feeling of what they need to learn to complete their contribution. It is important to notice that in the case of our experiment, we tried to remove the programming language barrier by directing newcomers to projects they could handle.

Participants also mentioned that the portal facilitated them **finding a task to start with**. The difference was mostly evidenced in analyzing JabRef participants because JabRef bugs were tagged as easy, but there was no information about it on JabRef's website. Information about LibreOffice's easy hacks, on the other hand, is widely available in LibreOffice developers' wiki.

*"I decided to take a look at the bugs listed as easy at FLOSScoach. I decided to take a deep look at this one https://sourceforge.net/p/jabref/bugs/960/, and I chose it because it seemed something simple and that I would handle..."* – FC02_UTF

The same participant confirmed it during the debriefing session:

*"[to choose a task] I went to FLOSScoach and tried to identify the simplest one"* – FC02_UTF (debrief)

*"It was a little hard to find a task that I understood. I started from the tool [FLOSScoach], then I tried to find using Bugzilla by using the link provided at*

*FLOSScoach. After some time, I went back to FLOSScoach and chose one from that list."* – FC07_UTF

It is important to notice that this list of tasks was filtering the tasks from the projects' issue tracker. To provide such a list, we had to have a list of issues that community members tagged with specific keywords.

#### *4.2.3.2 Social Interactions: no issues and no improvements*

Unfortunately, we noticed only a small number of social interactions between newcomers and communities during our study. We had two participants using Bugzilla to discuss the issue and one participant that joined IRC sessions mailing list and used the issue tracker to clarify these doubts.

Among the three people who interacted with the community, one participant explicitly mentioned following FLOSScoach's suggestion to interact with the community.

> *"I will explore FLOSScoach a little more. As it suggested, I will seek for help in the community. I will write a comment at the Bugzilla, to confirm my understanding about the bug and ask if I can assign the task to me."* – FC07_UTF

> *"I went to FLOSScoach and found out how to join the libreoffice IRC channel. I joined the #libreoffice-dev channel and waited until someone answer my question..."* – FC07_UTF

In addition to it, overall, the participants who interacted with the community did not report any reception incident, like lack of answer or improper/rude answers. The only issue was an unanswered question in IRC, which was explained as a time zone issue. One participant mentioned that the community was very receptive, and made him feel more comfortable in case he needed to contact them later on.

During the debrief session we tried to understand why the participants had not tried to interact with the community members. Five participants mentioned two reasons: English proficiency and fear. Four students mentioned the need to use English as the reason why they avoided approaching the community. Two people shared that they did not feel confident enough to send their questions, saying they were afraid to do so. The fear was related to lack of self-esteem and shyness, and can be verified in the following quotes:

> *"I did not try. I was afraid... of sending a newbie question, I don't know. Fear of being repressed. Because of the English, too. But, I think that even if it was in Portuguese I would not send."* – C07_UTF (debrief)

> *"I feel a little annoyed to talk about it, but I was afraid of the community. What if I ask something that is too simple. I think the community is something beautiful, that everybody is there to help, that they will welcome me because they need more contributors... I don't know, it is a little scary. What if I am*

*not good enough to do what was proposed... and... the community members do not like me... It is a problem with me."* – *C06_ UTF (debrief)*

One participant reported that the content of FLOSScoach was complete enough, and talking to the community was not necessary in his case. This participant effectively contributed to JabRef.

*"I did not need to talk to them. The tool was very clear. It is very easy, [the portal] is very good."* – FC02_UTF

As mentioned before, social interactions during this study were reduced. At a first glance, we noticed that in two cases the newcomers accepted the suggestions of the portal: one joined the IRC session twice and was directed to the mailing list, and another sent a message to the issue he was planning to work on. Only one newcomer who did not use the portal used IRC to try to solve his personal issues. Even knowing that more participants that used the portal interacted with the community, there was no evidence the tool fostered or encouraged communication as expected. A possible explanation is that the portal reduced the need for communication.

#### *4.2.3.3 Technical: barriers are still there, in both cases*

The diary and debrief sessions analysis revealed that portal use did not help overcoming technical-related problems, such as workspace setup issues. Issues with workspace setup and difficulties finding their way in the code were widely reported by both the control and study groups. This result aligns with the self-efficacy questionnaire results, which are detailed in Section 4.3. Furthermore, difficulty **understanding the architecture/code structure**, **understanding the code**, problems **finding the correct artifact fix an issue**, and many barriers inside **local environment setup hurdles** category, are also very evident and prominently represented in the diaries (in 12 out of 14). These barriers could not be diminished because the portal did not explicitly provide any new tool or mechanism to technically support newcomers.

Since the reports were written in parallel with the activity, and since most participants externalized their feelings about the problems, we could evidence to what extent the barriers influence newcomers' motivation. We found that 12 students reported issues **building workspace locally**, with six of these reports relating to frustration, irritation, or demotivation. We found that these feelings appeared in time-consuming activities with an unhappy end. In some cases, the bad feeling was reported when too much time was spent on fixing a single problem, mainly while also facing issues with missing dependencies:

*"The issues with the dependency are still there, so I decided to clone the repository again. I am feeling tired and frustrated"* – C04_UTF

*"I am still trying to build, because many errors occurred. The first error was because of the libgnome-vfsmm-2.6-dev. In the second try, I was trying to solve*

*the issues with Junit! Every time I need to run a 'make', it takes more than 40 minutes and, only today, I ran it 3 times. I was expecting to move forward, because so far I did not had time to look at the source code... It is frustrating."* – C05_UTF

In other cases, participants had to repeat the whole building process; including some participants who had to install another version (or even reinstall) their operating system:

*"I confirmed with a colleague issues regarding the version of my OS: it was not supported anymore, thus all repositories were offline. I needed to install a new linux distro that is supported."* – C06_UTF

*"I did not succeed on my last step (clone the repository again). Moreover, my OS presented some problems during the last process and I needed to format and reinstall ..."* – C04_UTF

A new barrier was identified. It was reported only by LibreOffice participants, and seemed to be project-specific: **lack of information about required disk space**. Some participants reported that more than 25GB were used at the end of the process. The main complaint about it was the lack of prior notification; the participants did not know in advance how much space they needed for LibreOffice.

*"It took some time, but I built it. Before, my hard disk was small and I faced some troubles... 10 GB [was required] to run without the debug, only for the release. To work on debug mode I needed 26 GB."* – C03_UTF (debrief)

*"[While cloning the repository] I had problems with disk space, I needed to clean some of my data..."*
[Some days later]
*"I will run a 'make clean' to clean what was done so far and run a 'make' once again... Another error. I have only 200MB available in my hard disk, I don't know what is the problem, but something is telling me that 'invalid pointer' and 'memory map' in the error message are related to my available space!"*
[The day after]
*"I will try again with 11GB available, I hope it works now"* – C05_UTF

The other very frequently reported barrier, which accompanied some bad feeling, was difficulty in finding the artifacts that should be amended. Participants used different strategies to approach this problem (facilities offered by the portal and by the project itself, or using previous knowledge), but the difficulty prompted them to mention irritation and demotivation.

*"I checked that bug again... The complicated thing is to find where to find the code I need to work on, because there are too many files and lines of code"* – C05_UTF

*"I think I will have to take a look at all the documentation. I have no idea of where the code that I need to change is."* – FC07_UTF

*"I really have no idea what I need to do now. I am stuck trying to find the code and there are only a few days left until the deadline..."* – FC07_UTF

*"Finding the right piece of code I need to change is too hard. I had already thought about how to code the bookmark stuff. But, finding the right place was the problem"* – FC06_UTF

*"I tried to find where to put the files to reference them inside the frame. I could change the icons that were already in place, but I did not figure out how to place a new icon"* – C02_UTF

These technical problems were more intense among the participants who were contributing to LibreOffice project. We can hypothesize some possible reasons. First, LibreOffice's codebase is larger and older than JabRef's, which can influence issues related to understanding code and finding the artifacts to change. Another possible relates to the technologies used to develop the systems. LibreOffice is developed mainly in C/C++ and depends on a huge set of libraries in specific versions, which are dependent on the Operating System the developers are using; this makes it more difficult to set up the workspace. Even providing several tutorial types, newcomers still face issues as reported. JabRef, on the other hand, uses gradle[60] to manage its dependencies (libraries). Once the newcomers found the available guidelines/tutorial in the project, the building was straightforward. The main issue for JabRef was finding the tutorial, which was not properly linked in the project page.

Technical issues were the most influential barriers for the newcomers studied, and they are the main reason why most students were unable to deliver their contributions in the defined period. The lessons that can be learned from this analysis are: (i) newcomers need to start with smaller and newer projects, which are easier to set up and provide more easily understood code; (ii) communities should give special attention to workspace setup issues and provide proper indications in the easy hacks, facilitating newcomers to find the artifacts that need to be changed.

We were expecting that it would be beneficial to provide ways for newcomers to search discussion archives, and point them to the communication channels. There is great potential for research that would explore offering proper support in these cases, and implementing tools to support newcomers in overcoming some of these barriers. By lowering such barriers, the contribution process will improve, and communities can benefit by receiving more contributions.

#### *4.2.3.4 Potential enhancements to the portal*

In order to prepare the portal for the second iteration, and to help in answering question Q1, we analyzed the diaries. We hoped to find possible suggestions for

[60] https://www.gradle.org

improving the portal, which we asked for during the debrief session with the participants who used FLOSScoach.

The most used feature of the portal, the contribution flow, was shown in a specific page, the "How to Contribute" page. However, some participants felt that it was not handy. They proposed to make the *flow accessible any time*, without having to navigate back it. Two of our participants who used the flow to guide their contribution offered this suggestion.

We received some other suggestions regarding the portal's organization. Two participants did not understand why *repeated information* appeared in different places. This repetition occurred because we built the first version of the portal by directly mapping the categories onto the portal. This led us to providing a "documentation" section; however, the information under documentation already appeared in other categories.

Another issue regarding the portal's organization concerned the *order of the categories* that was presented to the user. One participant suggested presenting the categories in the same order as they appear in the contribution flow. In addition, he suggested *avoiding the use of submenus*, making all possible categories only one click away.

The participants also suggested some features that might help newcomers choose a task and find the artifacts that they need to change in order to fix the issues. Two newcomers suggested the *recommendation of related information* to support their choice. One asked about the possibility of recommending other issues that are related to a given task, enabling newcomers to better understand the issue's context. The second suggestion was to indicate what part of the code newcomers should change to address the issue. These suggestions are very similar to the goal of some proposed initiatives in the literature, like in Wang and Sarma (2011) and Hipikat, by Cubranic et al. (2005).

We accommodated the suggestions related to the user interface and rearranged the portal structure for Iteration 2. We also updated barriers model according to suggestions about rearranging the structure. The recommendation of related information suggestions were not implemented, but is a possible future direction for this research.

### 4.2.4 Results for Iteration 2

Before discussing the research questions, we would like to highlight that, from the initial 27 subjects assigned to the control group (not using FLOSScoach), only 12 were considered (only 44%), as 15 did not complete the assignment (56%). From the 24 subjects assigned to use FLOSScoach 18 were considered (75%), only 4 did not complete the assignment (17%), and 2 decided not to use the portal (8%). This may indicate that FLOSScoach fostered or facilitated the contribution's completion.

The two subjects that received the credentials to use FLOSScoach and preferred not to use it declared they had no previous OSS development experience. When asked why they did not make use of the portal (after finishing the assignment), they reported that they tried to do it without any help, seeking to dive on their own into the open source process. One of them told us: "*I always try hard to find the information by myself, so I decided to not use FLOSScoach*" the other reported that "*I wanted to dive into the project... I planned to do whatever I could by searching the project page...*"

The results for Iteration 2 are in consonance with those obtained in Iteration 1. Thus, we did not present too much detail of the overlapping findings. In the following subsections, we briefly present the results of the Iteration 2 diary and debriefing session qualitative analysis.

#### *4.2.4.1 Process*

Regarding the contribution process, we could confirm that the portal played an important role orienting the newcomers and organizing the information they need to take their first steps in the project. From the 12 participants who were not using the portal, seven (58.33%) reported lack of orientation:

> *"When I accessed the project page (http://jabref.sourceforge.net/) for the first time, it confused me"* – *C07_ USP*

> *"After lurking the mailing list archives I am still without clues on how to contribute to the development."* – *C05_ USP*

> *"I spent some hours and other hours trying to find out what is the right path newcomers need to walk."* – C06_USP

> *"The information I found in the project website are long and confusing. I felt really lost and concerned."* – C14_USP

The participants who used FLOSScoach felt more confident and oriented during their first steps. Eight newcomers highlighted that in their diaries, for example:

> *"Although I got lost regarding understanding the code, I had almost no doubts about the process itself: downloading, find 'bugs', fix and submit the code. For me, the task was facilitated mainly by two factors: 1 - Presentation of necessary information only (searching and filtering information would require too much effort); 2 - organization of information (step by step guide)"* – FC01_USP

> *"[FLOSScoach] brought the facility of understanding how the contribution process is, the links to information about forks, pulls, git commands, tips to send a commit etc."* – FC10_USP

Presenting the technical skills needed and the technologies used by the project made newcomers aware of what they needed to learn to contribute, and it enabled the participants to find proper information before facing the barrier. We consequently

suggest that it lowered the barriers **knowledge on technologies and tools used** and **knowledge on versioning control system**. Due to his lack of knowledge in Git, for instance, one subject checked the skills needed using a portal tutorial:

> *"... Searching and reading some guides about Git, because I know the basics only. The guide provided by the FLOSScoach was really useful..."* – FC08_USP

We can say that, in general, newcomers felt oriented by FLOSScoach. Subjects mentioned benefits such as suggestions for finding a bug to start, links that point to the correct information, and the suggested contribution flow.

#### *4.2.4.2 Social*

With a larger sample and different profiles, we could notice a higher number of subjects interacting with the community. It was very common to find mentions to IRC chats, mailing list messages and entries in the issue tracker. Both subjects using and not using FLOSScoach interacted with the community more often, and in more ways.

Even noticing a high response rate, in some cases the newcomers did not receive any answer from the community, which lead to frustration.

> *"This was supposed to be a moment of excitement and anxiety, but the silence of the community made me feel anesthetized. On the one hand, this is an activity that does not bother me, on the other does not lead me to an effusive state as I imagined at the beginning of the task. I am blasé!"* – C06_USP

> *"I asked this question after searching for a certain period. I posed the question politely, as required by the community. I received no answer..."* – FC09_USP

> *"Let's see how friendly these guys are. I added a ticket, asking for help..."*
> *[4 days later]*
> *"I did not receive any answer"* – FC10_USP

An important finding is that those newcomers who received response reported no cases of receiving improper answers. All the subjects that communicated with the community mentioned that they received welcoming messages and proper orientation.

> *"I surprisingly received an answer to my email few hours after sending it to the community"* – FC15_USP

> *"The community members were very friendly and welcoming. I will probably become a regular contributor."* – FC17_USP

> *"I joined the IRC channel of the LibreOffice developers, and they helped me a lot."* – C10_USP

A subject mentioned another benefit of using FLOSScoach; that the message template provided by the portal was helpful. He reported that the template helps newcomers to be clearer, more concise, and to reduce the shyness:

*"I liked the message template, showing how to introduce myself and to present the problems I am facing. Even having proficiency in English, I did not know the more polite way of asking for help. This example helped to be clear, concise to present the message objective, and also to reduce the shyness"* – FC01_USP

Even with some indications that the portal supported social interactions, finding better ways to foster the communication with the community is a topic that deserves further investigation.

#### *4.2.4.3 Technical issues*

The findings regarding technical issues are very similar to those presented beforehand. Participants recurrently mentioned issues setting up the local workspace, finding the right place to fix the bug, and understanding code. Once again, technical issues were reported, along with bad feelings (mainly frustration and irritation).

A different finding concerns providing ways for newcomers to search the discussion archive. Only one participant mentioned using the embedded search. He reported that he *"...succeeded thanks to this e-mail https://forums.wxwidgets.org/viewtopic.php?f=23&t=26587, that I found using this page http:/flosscoach.com/index.php/communication-audacity"* – FC05_USP.

Regarding the differences related to codebase size mentioned in Iteration 1, we could not confirm the findings. The issues related to setup and understanding code were mentioned for almost all C/C++ projects (5 out of 6 objects of study), regardless the project's size. The impact of language still needs to be further investigated, since JabRef participants complained less than other participants and 4 had contributions successfully accepted.

To conclude, we so far could not find any strong evidence that FLOSScoach can offer technical problem support. This quote from FC17_USP, from the debrief questionnaire, evidences this:

*"FLOSScoach is really interesting... It was a good starting point, that helped me learning the etiquette and the process, but it did not help me on the technical development problems" – FC17_USP*

Reinforcing what was said in Iteration 1, we advise communities to take special care of workspace setup, and to provide ways to help newcomers find their way in the code.

#### *4.2.4.4 Potential enhancements to the portal*

As in Iteration 1, we analyzed the data we gathered, searching for suggestions to enhance the portal. We anticipate that there was no suggestion or complaint that required changes in the barriers model. However, FC02_USP complained about the

portal's organization, saying that contribution flow should be the main guide for the portal, instead of using the portal sections.

FC17_USP suggested some indications of "*how to choose a good easy bug to start with ... In addition to pointers to the bug, there are no hints on where to look to help me deciding.*" Also related to choosing a task to start with, and in consonance with the Iteration 1, two newcomers recommended providing what part of the code the newcomers should alter to address the issue.

Some other participants suggested some changes and new features related to making the portal a more collaborative environment. Two recommended the creation of one forum per project, and one of them (FC01_USP) justified it saying: "*thus, you can be aware of other newcomers who are working in the same project and talk to them inside the portal.*"

Also related to collaborative features, FC13_USP suggested creating reputation and tagging mechanisms for the posts. This would make the "*newcomers aware of the quality of the posts, based on rates, number of visualizations, tagging etc.*" FC15_USP reinforced this suggestion by mentioning the possibility of fostering the participation of community members, who could endorse the information available in the portal and answer forum posts.

Iteration 2's suggestions did not affect the barriers model; however, they are valuable ideas for enhancing the portal. In the context of this thesis, we did not implement the proposed improvements.

### 4.2.5 Barriers lowered by FLOSScoach use

In this section, we present a summary of the barriers that we determined were lowered when newcomers used the FLOSScoach portal.

Before presenting it, we should mention that we could not evidence any new barrier that emerged due to the use of the FLOSScoach portal. The few issues reported by the participants were classified as enhancement opportunities, since they were mostly related to the way FLOSScoach presented information.

We understand that FLOSScoach's main contribution was better orienting newcomers by organizing already provided strategies and documents. This brought direct results in terms of reducing newcomer orientation barriers as well as some documentation barriers. In Table 4.4, we summarized the subjective evaluation of the barriers that could be lowered by FLOSScoach use.

Table 4.4. Summary of barriers lowered by the use of FLOSScoach.

| **Categories/Subcategories/Barriers** | | | | **Subjective evaluation** | **Explanation based on the qualitative data** |
|---|---|---|---|---|---|
| Newcomers' characteristics | Newcomers' previous knowledge | Lack of knowledge in project process and practices | | *Strongly decreased* | This barrier was lowered by explicitly presenting the steps newcomer should follow and linking the steps to proper documentation/information |
| | | Lack of technical background | Knowledge on technologies and tools used | *Slightly decreased* | The barriers under this category were lowered by presenting to the newcomers what are the technologies and tools they will need to use to contribute to the project. The way to reduce this barrier depends on the newcomers' attitude and knowledge. However, we verified that it is possible for the community to take some actions that benefit newcomers. |
| | | | Knowledge on versioning control system | *Slightly decreased* | |
| | | | Choosing the right development tools | *Slightly decreased* | Newcomers mentioned they chose (or changed) the development tools used, based on the information provided in FLOSScoach. |
| | Newcomers' behavior | Newcomers' communication | Shyness | *Slightly decreased* | A newcomer mentioned that the suggested message template help newcomers feel more confident and less shy to send a message, since the example shows how to send clear, concise, and objective messages. |
| | | | Not sending a correct/meaningful message | *Slightly decreased* | |
| Newcomers' orientation | Finding a task to start with | | | *Decreased* | Newcomers reported the benefit of presenting a filtered list of tasks tagged as easy for newcomers by the community. We are aware that such feature depends on the community action. |
| | Poor "How to contribute" available | | | *Strongly decreased* | We directly addressed this issue by presenting the information organized and by providing a general flow of contribution |
| | Newcomers don't know what is the contribution flow | | | *Strongly decreased* | Newcomers made use of and reported the contribution flow presented in the tool to contribute |
| Documentation problems | Information overload | | | *Decreased* | The number of reports mentioning problems too much documentation, and talking reporting lost feelings evidence the reduction of this barrier |
| | Spread documentation | | | *Decreased* | |
| | Outdated documentation | | | *Slightly decreased (few evidences)* | Newcomers using the portal did not report such issue. However, newcomers who did not have access to the portal faced these barriers, causing further problems. |
| Technical hurdles | Change request hurdles | Lack of information on how to send a contribution | | *Slightly decreased* | Newcomers who reached this point reported they used the information provided by FLOSScoach to complete (send) their submission. |
| | | Issue to create a patch | | *Slightly decreased* | |
| | Local environment setup hurdles | Finding the correct source | | *Slightly decreased* | Newcomers using the portal did not report such issue. However, newcomers who did not have access to the portal faced the issue (caused by outdated documentation and information overload). |

Although not mentioned specifically in the table, some social barriers could be softened by reducing the need for community interaction, as reported by one participant: *"I did not need to talk to them. The tool was very clear. It is very easy, very good."* – FC02_UTF

We are not able to say that FLOSScoach portal use eliminated the barriers. However, we observed that these barriers were somehow reduced, since they were reported by the control group and not by the group using the portal.

In Iteration 2, we collected more evidences related to the barriers lowered by portal use; and, moreover, we found two other lowered barriers. The first relates to social interaction with the community. A newcomer reported the example of a message that helped with the first community interaction, lowering the newcomers' **shyness** barrier and helping them to **send meaningful/correct message**.

The second additional barrier lowered by portal use concerned searching the mailing list archive, which supports newcomer finding help based on previously discussed topics. In this case, a search using such features enabled a newcomer to solve a workspace setup issue, as reported in the Section 4.2.3.3. The search feature thus supported the newcomers' in overcoming a **local environment setup hurdles**, and reducing **newcomers' communication issues.**

#### *4.2.5.1 A quantitative look at the barriers identified by the participants*

Complementary to the qualitative results presented in the previous sections, we observed which barriers the participants of the study reported. We did that by coding the diaries and feedbacks according to the barriers model, but also looking for any new barrier that should be reported. Our goal was not to check the importance or the level of influence of the barriers, but to determine if there were noticeable differences between the barriers the participants who used the tool reported, versus those reported by the control group participants.

In Table 4.5 we present the barriers the participants reported in their diaries during the assignment. In the last four columns, we present the number of participants who reported the barrier in each group for Iteration 1 and Iteration 2, and the percentage according to the total of participants per group. We present only the barriers that were reported at least once, omitting those that were not mentioned.

In Iteration 1, the number of barriers reported by the participants who used FLOSScoach (Case) is smaller than those reported by the participants who did not use it (Control). Whereas the control group reported 27 barriers, the case group reported 20 barriers. The largest difference relates to orientation and finding proper information barriers (mainly under **Newcomers' orientation** and **documentation problems**). On the other hand, participants of both groups overwhelmingly reported **technical hurdle** barriers, which were the most reported barrier type.

Table 4.5. Barriers reported by participants during the assignment.

| Category/Subcategory | | Barrier | Iteration 1 | | Iteration 2 | |
|---|---|---|---|---|---|---|
| | | | Control | Case (using FLOSScoach) | Control | Case (using FLOSScoach) |
| | | Number of participants | 8 | 6 | 14 | 18 |
| Newcomers' orientation | | Newcomers don't know what is the contribution flow | 5 (63%) | **0 (0%)** | 6 (42.9%) | **0 (0%)** |
| | | Poor "How to contribute" available | 4 (50%) | **0 (0%)** | 2 (14.3%) | **0 (0%)** |
| | | Finding a task to start with | 6 (75%) | 2 (33%) | 11 (78.6%) | 3 (17.6%) |
| | | Finding the correct artifact to fix an issue | 5 (63%) | 4 (67%) | 2 (14.3%) | 7 (41.2%) |
| | | Finding a Mentor | - | - | 1 (7.1%) | 1 (5.9%) |
| | | Outdated list of bugs | **0 (0%)** | 1 (17%) | 2 (14.3%) | 4 (23.5%) |
| Newcomers' characteristics | | Lack of domain expertise | - | - | 2 (14.3%) | 2 (11.8%) |
| | Newcomers' behavior | English level | 2 (25%) | 1 (17%) | 1 (7.1%) | **0 (0%)** |
| | | Lack of commitment | 1 (13%) | **0 (0%)** | - | - |
| | | Shyness | 2 (25%) | **0 (0%)** | 1 (7.1%) | **0 (0%)** |
| | Newcomers' previous knowledge | Knowledge on versioning control system | 1 (13%) | **0 (0%)** | 0 | 1 (5.9%) |
| | | Knowledge on technologies used | - | - | 1 (7.1%) | 1 (5.9%) |
| Reception issues | | Receiving answers with too advanced/complex contents | **0 (0%)** | 2 (33%) | - | - |
| | | Not receiving an answer | - | - | 2 (14.3%) | 2 (11.8%) |
| | | Delayed answers | - | - | **0 (0%)** | 2 (11.8%) |
| Documentation problems | | Information overload | 2 (25%) | **0 (0%)** | 1 (7.1%) | **0 (0%)** |
| | | Outdated documentation | 1 (13%) | **0 (0%)** | 2 (14.3%) | **0 (0%)** |
| | | Spread documentation | 2 (25%) | **0 (0%)** | 2 (14.3%) | **0 (0%)** |
| | Lack of documentation | Documentation in general | 2 (25%) | **0 (0%)** | 4 (28.6%) | 4 (23.5%) |
| | | Code documentation | 2 (25%) | 2 (33%) | 1 (7.1%) | 1 (5.9%) |
| | | Documentation on setting up workspace | | | 2 (14.3%) | **0 (0%)** |
| | | Design documents | 3 (38%) | 2 (33%) | **0 (0%)** | 1 (5.9%) |
| | | Code comments | 2 (25%) | **0 (0%)** | 1 (7.1%) | 1 (5.9%) |
| | | Documentation on code structure | 2 (25%) | **0 (0%)** | | |
| Technical Hurdles | Code/architectural hurdles | Bad code quality | 1 (13%) | **0 (0%)** | 1 (7.1%) | 1 (5.9%) |
| | | Bad design quality | - | - | **0 (0%)** | 1 (5.9%) |
| | | Codebase size | 2 (25%) | 2 (33%) | 3 (21.4%) | 3 (17.6%) |
| | | Outdated code | 1 (13%) | 1 (17%) | - | - |
| | | Understanding the code | 3 (38%) | 3 (50%) | 2 (14.3%) | 1 (5.9%) |
| | | Understanding the architecture/code structure | 4 (50%) | 2 (33%) | 2 (14.3%) | 1 (5.9%) |
| | Change request hurdles | Issues to create a patch | 1 (13%) | 1 (17%) | 1 (7.1%) | 2 (11.8%) |
| | | Delay to get contribution accepted/reviewed | **0 (0%)** | 1 (17%) | 1 (7.1%) | 1 (5.9%) |
| | Local environment setup hurdles | Finding the correct source | 2 (25%) | 1 (17%) | 1 (7.1%) | **0 (0%)** |
| | | Building the workspace locally | 8 (100%) | 3 (50%) | 10 (71.4%) | 8 (47.1%) |
| | | *Lack of information about required disk space*** | 2 (25%) | 2 (33%) | - | - |
| | | Platform dependency | 3 (38%) | 1 (17%) | 4 (28.6%) | 1 (5.9%) |
| | | Library dependency | 4 (50%) | 1 (17%) | 9 (64.3%) | 5 (29.4%) |

*** new barrier identified*

A similar pattern was observed in Iteration 2. The number of barriers mentioned by the newcomers who used the portal was 23 in this case, versus 28 reported by the control group. We can see a noticeable difference in barriers related to **newcomers' orientation** and **documentation problems.**

Analyzing the results of both iterations, we see results that align with the results presented earlier. Although the portal supported the newcomers to overcome contribution process barriers, we could not observe noticeable improvements related to technical barriers. Other interesting information from the table deserves highlighting. The barriers **information overload, outdated documentation** and **spread documentation**, under documentation problems, were not mentioned by any of the 24 participants that used FLOSScoach. The same behavior was observed for the barriers **newcomers don't know what is the contribution flow** and **poor "How to contribute" available**, under **newcomers' orientation**. In addition, more than 70% of the control group participants mentioned the barrier **finding a task to start with**, whereas only 21% of the participants who used FLOSScoach mentioned it.

Some barriers related to **newcomers' characteristics**, mainly those related to behavior, were not mentioned, or were mentioned by only one newcomer. Since a newcomer mentioned that the portal reduced the need to contact the community, and another mentioned that the examples presented in the portal reduced shyness and helped in writing clearer messages, it would be interesting to further explore these barriers.

### 4.2.6 Summary

We could see that FLOSScoach provided proper guidance to newcomers during the contribution process, benefitting them in different ways. Some used it as a definitive guide, relying on the suggested flow to proceed with their contribution, then following the flow, and then clicking on the options the portal offered before taking any other action. Other participants used the portal as a quick reference guide, to which they could come back at any time and easily find the information they needed to finish a task or to overcome a problem.

The tool served as a compass for the newcomers to orient themselves in an OSS software landscape. We found that by simply organizing the information according to the barriers model, as well as including some data obtained from the practitioners, the tool could help newcomers overcome a set of barriers. In the post-assignment surveys, newcomers highlighted the portal's step-by-step contribution flow and its organization of information.

Even though it provided good guidance, there were lower benefits when using FLOSScoach to overcome social and technical barriers. However, some newcomers mentioned how they used the portal to overcome some of them. Moreover, by providing such guidance, we helped newcomers feel more confident and fight harder to overcome the barriers, which possibly increased success rate.

## 4.3 Does the use of the portal impact participants' self-efficacy?

Self-efficacy is a measure of the confidence in the participants' perceived ability to perform a task, which can impact one's actual ability to complete a task (Bandura, 1986). It is correlated with the willingness to stick with a learning task, and has been studied in computer science education (Cassidy and Eachus, 2002) and OSS (Davidson et al., 2014) contexts. Based on previous work that applied self-efficacy in OSS research (Park, 2008; Davidson et al., 2014), we prepared a questionnaire with 10 items related to self-efficacy of contributing to OSS on a five-point Likert scale, as presented in Table 4.6. The items of the questionnaire were based on the previous work conducted by Davidson et al. (2014) and Park (2008), who applied self-efficacy in research related to OSS joining process.

We asked the participants to answer it immediately before they started their assignment, and also immediately after concluding it. We aimed to discover whether the person had success performing the tasks (resulting in an increase in self-efficacy), or faced unexpected problems or failures (resulting in a decrease in self-efficacy)(Smith et al., 2006).

Table 4.6. Items on self-efficacy and interest toward OSS activities.

| Sentence |
|---|
| 1. I feel comfortable asking for help from the community using electronic communication means |
| 2. I can write my doubts and understand answers in English |
| 3. I am good in understanding code written by other people |
| 4. I have pretty good skills to write and change code |
| 5. I feel comfortable with the process of contributing to an Open Source project |
| 6. I think that contributing to an open source software project is an interesting activity |
| 7. I feel I can set up and run an application if a set of instructions is properly given |
| 8. I am pretty good on searching for solutions and understanding technical issues by myself |
| 9. I can choose an adequate task to fix if a list of tasks is given |
| 10. I can find the piece of code that need to be fixed given a bug report presenting the issue |

### 4.3.1 Results for Iteration 1

To verify how the use of FLOSScoach influenced participant's self-efficacy, we analyzed the variation of the pre- and post-study questionnaire answers. To conduct our analysis, we dismissed the answers from FC5_UTF, the participant who was supposed to use the portal, but declared that he did not.

Figure 4.5 presents the sum of all answers given by each participant to self-efficacy questionnaire, comparing pre- and post-study results. In Figure 4.5(a), the data is related to the participants who used the portal to contribute. In Figure 4.5(b), we show the results for the participants who did not have access to the portal. In both figures, we highlight the project to which the participants were asked to contribute (presenting dashed boxes), and we represent the last step the participant reached: (1) workspace setup; (2) finding a task; (3) finding the piece of code to work on; (4) fixing the bug/submitting the fix; (5) fix accepted.

As shown in Figure 4.5 (a), the self-efficacy of six out of seven participants who used the portal increased. This shows that most of the participants finished the study more confident than when they began it. The participant FC6_UTF is the one that did not make use of the portal, and also presented an increase in self-efficacy. This increase can be explained by his previous experience, which influenced the way he approached the barriers. For participant FC1_UTF, we can see an increase. This can be explained by the fact that this participant was able to very quickly contribute to the project.

The decrease observed on FC7_UTF self-efficacy was analyzed in detail. His score was influenced by the answers to Q6 and Q10. On Q6, the values pre- and post- study are 5 and 3, while for Q10 the values were 4 and 1, respectively. These two questions explain 5 out of the 8 points difference on pre/post analysis. A possible explanation for this is that his self-confidence decreased. During the debrief session, this participant was very upset about his performance,

because he was stuck on finding the right artifact to change. He reported spending 2 weeks to find it, and could not complete the assignment, since he was not able to test his changes.

Analyzing Figure 4.5(b), we can see a slightly different scenario. Most of the participants that did not have access to the portal (4 out of 7) presented a decrease in self-efficacy. In accordance with the results presented by Davidson et al. (2014) in their study of older adults, we attribute the decreases in self-efficacy to the idea that "you don't know what you don't know." If participants experienced unexpected barriers, their self-efficacy significantly decreases.

Notably, two of the participants whose self-efficacy increased were contributing to JabRef (C1_UTF and C2_UTF). Therefore, all the participants assigned to contribute to JabRef increased their self-efficacy score. We attribute this behavior to the complexity of the project, which exposed the newcomers to lower barriers, mainly related to workspace setup and understanding code.

To better understand the different behaviors observed, Figure 4.6 shows the median of the answers per question. We can thereby observe trends that encouraged self-efficacy up, and others that pulled it down. First, in consonance with the diary entries, the scores for questions related to the OSS contribution process (Q5 and Q6) increased for the participants that used the portal, and decreased for those who did not. The decreasing trend observed for Q10 represents the newcomers' difficulty in finding the artifact they need to change in order to work on a selected task, reported by newcomers in both groups. Another interesting behavior is observed for Q8 and Q9 (mostly on Q9), showing their self-confidence to choose a task to work on, even when not using the portal. We did not find any explanation for the variations in Q1 and Q2, since there was a small number of social interactions with the community.

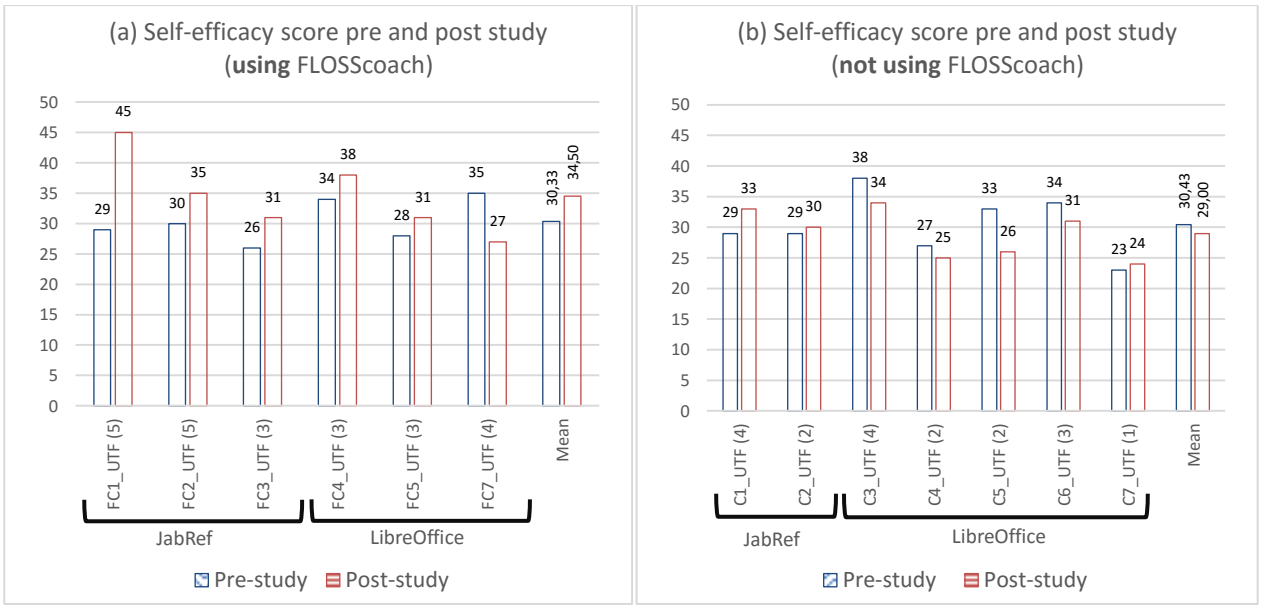



Figure 4.5. Self-efficacy results per subject (pre and post questionnaires).

We also conducted a Wilcoxon signed-rank test to verify whether the differences observed in the questions were statistically significant or not. However, we found no significant difference for any of the questions.

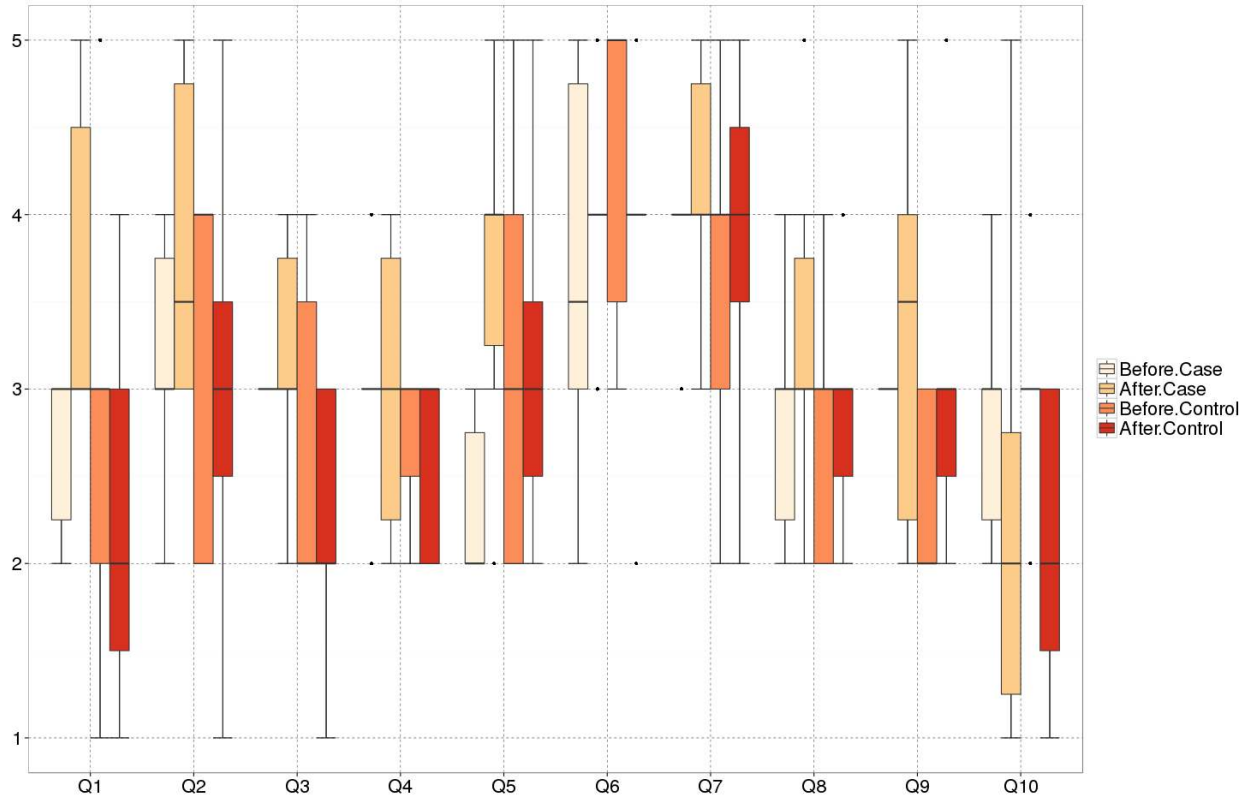



Figure 4.6. Self-efficacy results per question.

### 4.3.2 Results for Iteration 2

Following the same method as in Iteration 1, we applied the self-efficacy questionnaire before and after the students received the assignment. In Figure 4.7 and Figure 4.8, we present the results of self-efficacy pre- and post-study for the group who used the portal and the group who did not use it, respectively. In both figures, we highlight the project to which the participants were asked to contribute (presenting dashed boxes), and we represent the last step the participant reached: (1) workspace setup; (2) finding a task; (3) finding the piece of code to work on; (4) fixing the bug/submitting the fix; (5) fix accepted.

As can be observed in Figure 4.7, the self-efficacy of 11 participants who used the portal increased, two remained the same, and 5 decreased. Most of the participants finished more confident than when they started the study. It is possible to observe that none of the participants with a decreasing behavior achieved a further step than finding a bug to work with. Another aspect

that appears to be relevant is that all the participants that worked on the Vim project presented a self-efficacy decrease.

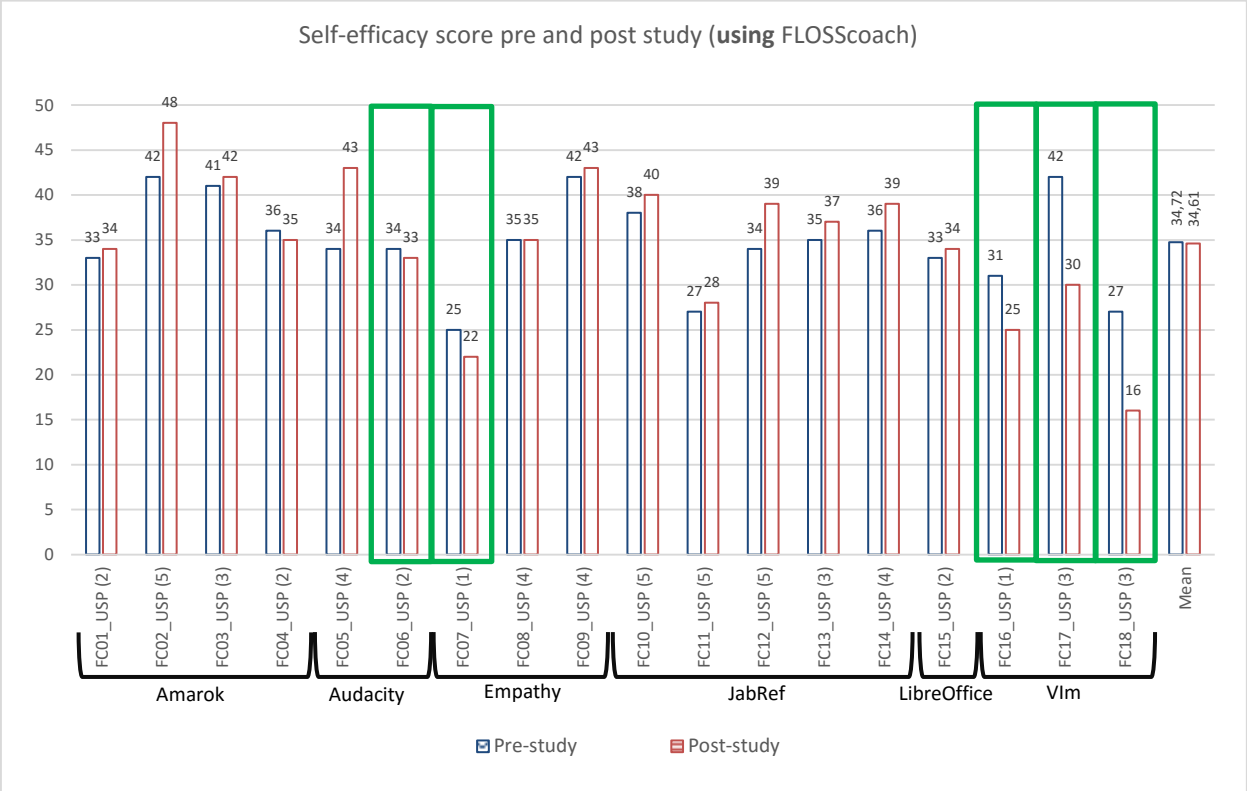



Figure 4.7. Self-efficacy scores for participants who used FLOSScoach.

We can see three decreases (for participants FC16_USP, FC17_USP and FC18_USP), and two smaller decreases (FC6_USP and FC7_USP). Regarding FC16_USP's decrease, we found two main axes that contributed to the decrease: social interactions (Q1 and Q2), in which we observed a decrease of 5 points compared to the pre-study; and code issues, contributing to a 7-point decrease. More specifically, for the code issues we observed that the self-efficacy reported in questions Q3, Q4, and Q10 decreased 1, 4, and 2 points, respectively. This aligns with the diaries and answers to the debrief questionnaire. In his diary, it was possible to find evidence of changing the selected task at least thrice, trying to understand it and find possible solutions (and complaining about his difficulty in understanding the code).

In his diary and in debrief, FC17_USP complained that the bugs were not classified by difficulty level and the community reception was poor. He tried to interact with the community via IRC and e-mail (trying to achieve a mentor designated by the project), but received no answer. In addition, in debrief he highlighted problems in understanding the code and lack of knowledge of the application as contribution barriers. Regarding his self-efficacy answers, the decrease behavior spread across all questions. However, the 5 points decrease in the last three questions (-1, -3 and -1, respectively) concerned finding a task and where to address a bug.

For FC18_USP we noticed in all questions a decrease of 1 or 2 points. We could not find any evidence for this decrease in the debrief questionnaire. Observing the diaries, we saw that the participant spent a large amount of time building his local environment, and after that he could not find the right place to address the issues he selected. This lead him to changing the chosen bug twice. However, the participant presented no specific complaint or issue. We contacted the participant to investigate the reasons. We found that a single cause impacted his score as a whole. In his case, the problem was finding the right artifact to work on. This can be evidenced in the quote:

> *"As I faced many problems to find the buggy feature, and due to external factors (time), my self-efficacy was negatively impacted."*

For the participants FC6_USP and FC7_USP, we found complaints related to dealing with a large project combined with the short time period allowed to contribute, which caused problems in understanding the code. Looking at their self-efficacy results, we found that both presented a decrease in the two last questions, which related to finding a task to work with and the right artifact to fix the bug.

For the participants who had no access to FLOSScoach (control group), we observe an opposite behavior (Figure 4.8). The self-efficacy score of nine participants decreased after the study, whereas only four increased. Among the four participants with an increasing self-efficacy, two finished their assignment with their contribution accepted by the community. We did not find any pattern for the scores of the participants with an increasing score. The most interesting behavior was evidenced for C11_USP, presenting higher self-efficacy score in 8 out of 10 questions, while the other two scores remained the same as in the pre-study.

Significantly, five other subjects submitted or had their fixes accepted (C4_USP, C6_USP, C7_USP, C10_USP and C13_USP), yet presented a decrease in their self-efficacy scores. Analyzing their answers, we found that there is a decrease for each of these subjects in the code-related questions, as well as in the contribution process questions for three subjects.

Also notable is the large decrease in C2_USP's score, which drops by 20 points. By checking the differences in each question, we observed that all the scores decrease, but the highest differences are in questions Q5, Q6, and Q7, with a 3-point decrease each. These questions concerned confidence in the contribution process and in workspace setup. This second point was highlighted by the participant in his diary and in debrief. We tried contacting the participant to better understand the problem, but did not receive an answer.

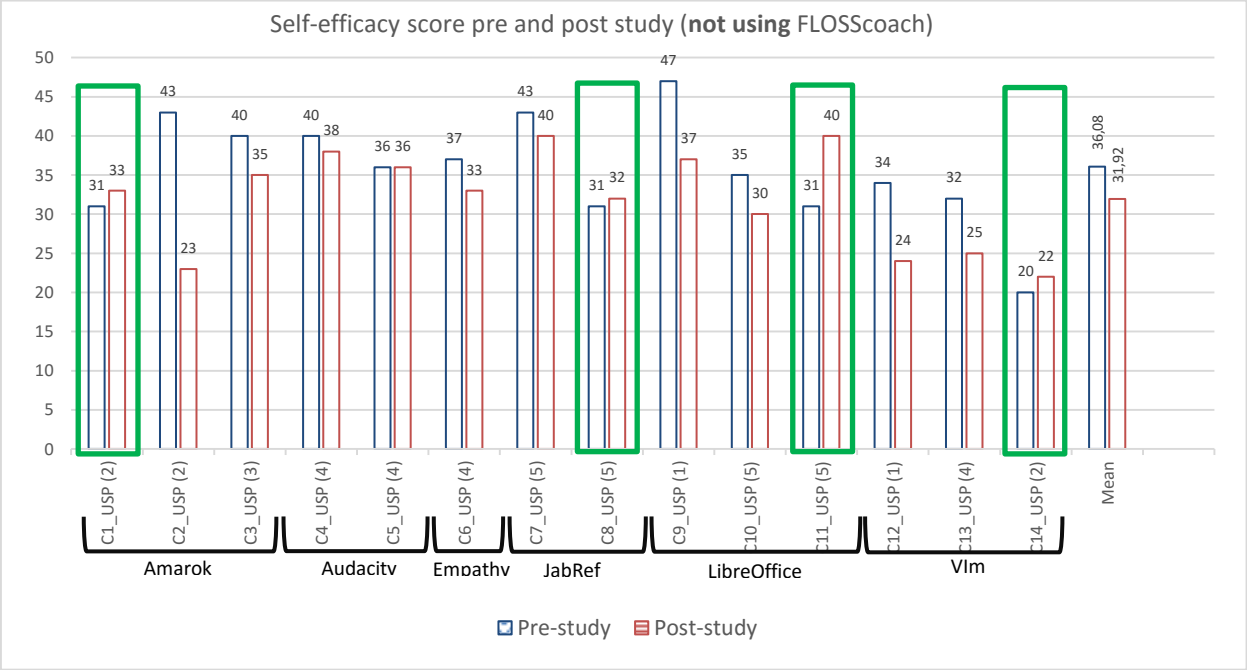



Figure 4.8. Self-efficacy scores for participants of the control group.

As in Iteration 1, we presented the median of the answers per question to provide some explanation for the variation of self-efficacy. In Figure 4.9 and Figure 4.10, we can observe trends that pushed self-efficacy up and others that pulled it down. First, in consonance with the previous iteration and the diary entries, the scores for questions related to the contribution process (Q5 and Q6) remained equal and increased for the participants that used the portal, whereas it decreased for those who did not. Another increasing trend observed in Figure 4.9 relates to easiness in finding a task to work with (Q9). The difference is less pronounced, but can indicate a possible facility in finding the right task to work with for the participants using FLOSScoach portal.

The decreasing trend observed for Q10 represents the difficulty in finding the artifact newcomers need to change in order to work on a selected task, which was reported by newcomers in both groups. Another interesting behavior observed for Q8 and Q9 (mostly on Q9) shows participants' self-confidence in choosing a task to work on, even when not using the portal. A possible explanation for the decrease in Q1 score is the number of answered questions reported in this iteration (5). However, we believe that this is not the sole reason for this, and that this point deserves further future investigation.

As in Iteration 1, there is some indication of technical problems in the self-efficacy results, however they are less pronounced. This can be observed in the variations of Q3 in Figure 4.9 and Q4 in Figure 4.10. There was no evidence of difficulty finding the artifact(s) that needed to be changed to fix a bug, in contrast to Iteration 1's clear difference (refer to Figure 4.6).

We conducted a Wilcoxon signed-rank test to verify whether the differences observed in the questions were statistically significant or not. For the group who used FLOSScoach, we found that

there was a decrease in Q1 (p=0.021) and an increase in Q6 (p=0.062). For those who did not use the portal, we found a statistically significant decrease in Q4 (p=0.042) and Q5 (p=0.025). In addition, we found that the difference in the total self-efficacy score significantly decreased for the participants who did not use the tool (p=0.025).

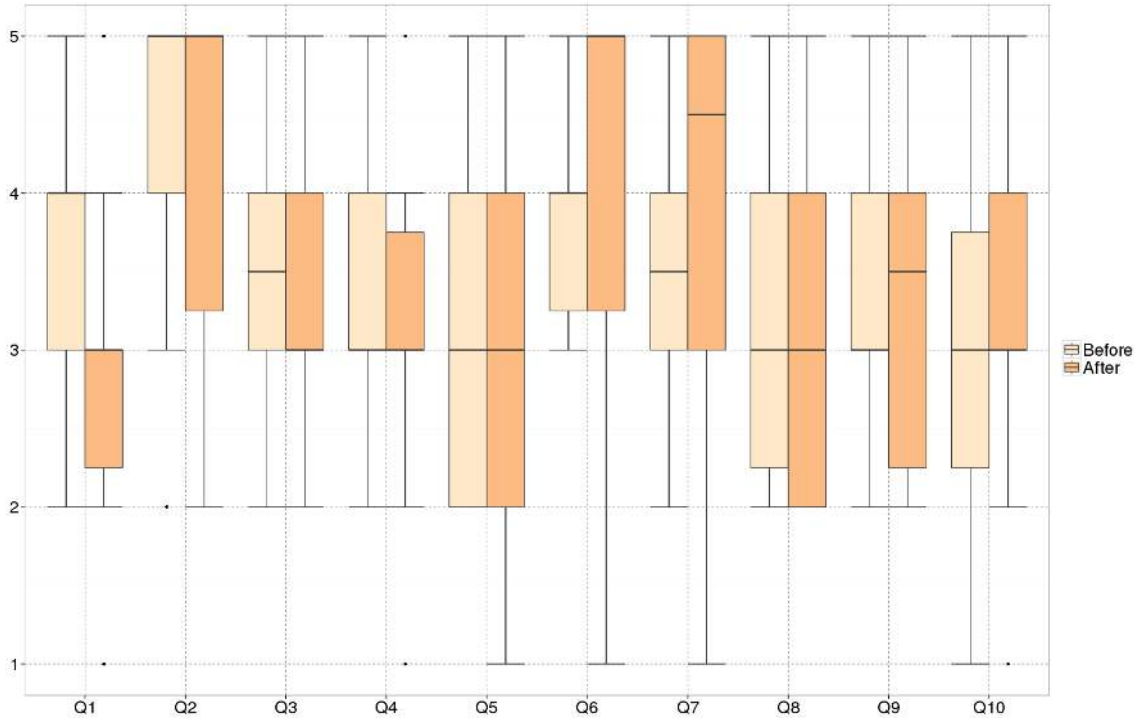



Figure 4.9. Median of answers per self-efficacy question (participants who **used** FLOSScoach).

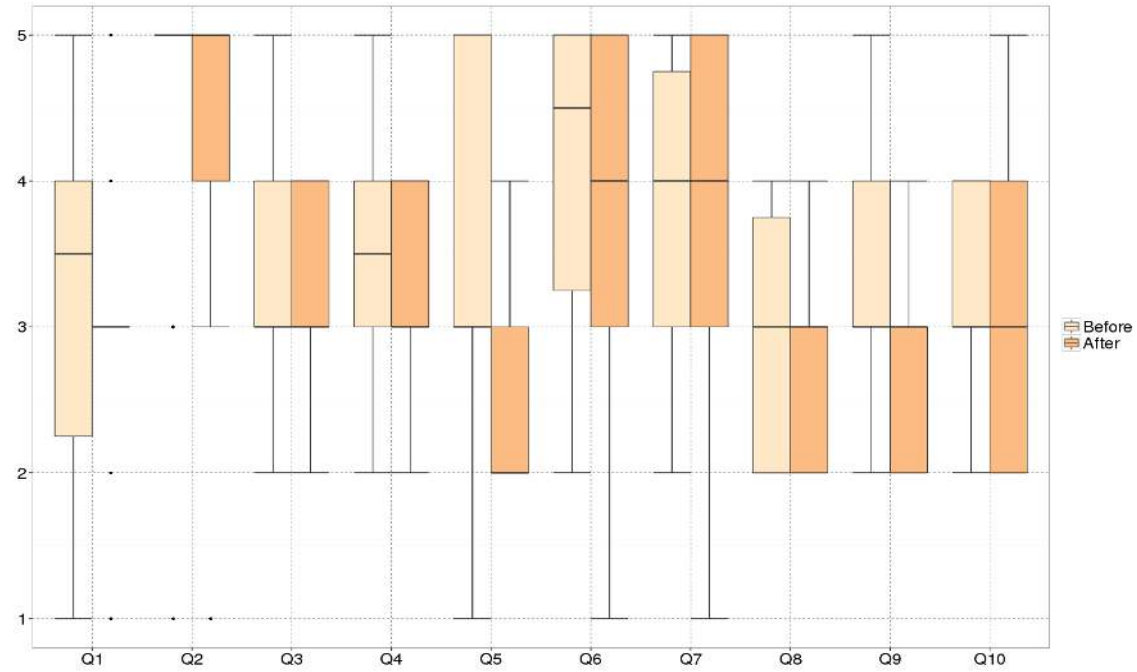



Figure 4.10. Median of answers per self-efficacy question (participants who **not used** FLOSScoach).

### 4.3.3 Summary

We found that the use of FLOSScoach may had positively influenced on newcomers' self-efficacy, mainly by making them more confident and comfortable during the OSS project contribution process. However, there was also some indication that FLOSScoach did not lower the technical barriers. These results are in line with the results obtained in the diaries analysis.

## 4.4 What is this portal's perceived usefulness, ease of use, and potential future use?

To answer these questions, we applied the Technology Acceptance Model (TAM) (Davis, 1989), which is a model inspired by the theory of reasoned action (TRA) (Fishbein and Ajzen, 1975). TRA asserted that attitude towards an action and subjective norms together impact behavioral intention, which in turn affects how people perform the action (King and He, 2006). TAM aims at assessing user perception about a technology's usefulness and ease of use, which are the basic determinants of a user's acceptance behavior (Laitenberger and Dreyer, 1998). Perceived usefulness is defined as "the degree to which a person believes that using a particular technology would enhance his or her job performance," and perceived ease of use refers to "the degree to which a person believes that using a particular system would be free of effort." (Davis, 1989) According to the theory of reasoned action, these factors strongly correlate to one's intention to actually use a technological innovation, if it is available, i.e., self-reported future use, as modeled in Figure 4.11.

TAM has been widely applied in technology assessment, producing reliable results when tested users have worked with the technology for some time. King and He (2006) report on the results of a meta-analysis of 88 published TAM studies supporting the instrument's validity and robustness with a wide range of applications. Laitenberger and Dreyer (1998) provide a good introduction to TAM adaptation for software tool acceptance measurement.

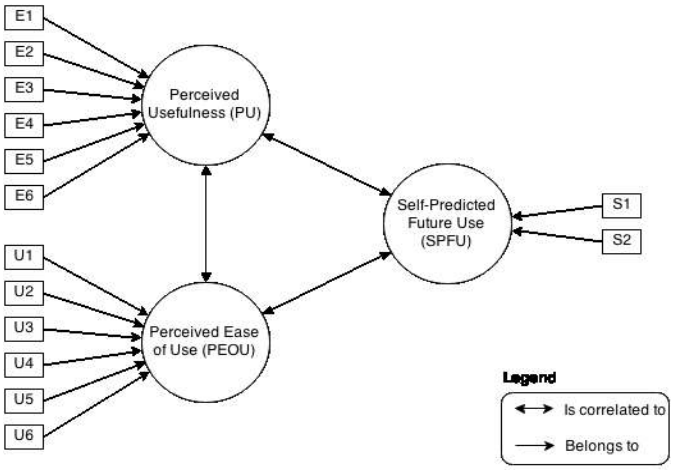



Figure 4.11. Model of usefulness, ease of use, and self-predicted future usage (TAM).

We found TAM a suitable starting point for developing an adapted measurement tool that could assess FLOSScoach's "ease of use" and "usefulness" in supporting newcomers' contribution process. Table 4.7 shows our TAM-based assessment model adapted from (Davis, 1989; Laitenberger and Dreyer, 1998; Babar et al., 2007; Vaz et al., 2013). There are sets of questions that measures each of the three main constructs: "perceived usefulness" (Ui), "ease of use" (Ei), and "self-predicted future use" (Si). We used the same items presented by Laitenberger and Dreyer (1998), based on the item set originally proposed by Davis (1989).

To adapt the TAM-based assessment instrument we: (i) used "FLOSScoach" as the object of the questionnaire; and (ii) named the process the questionnaire investigated: "newcomers contribution to OSS projects." We used a Likert scale to measure participants' perceptions, using a six-point semantic scale that asked for their degree of agreement with a statement about their perception ("extremely disagree" (1) to "extremely agree" (6)). In accordance with the study by Babar et al. (2007), we chose a 6-point scale with no neutral value available so subjects could more clearly express their tendency towards a positive or negative evaluation.

Table 4.7. Scale items for measuring usefulness, ease of use and self-predicted future use.

| | |
|---|---|
| **Perceived Usefulness (PU)** | |
| U1. | Using a portal like FLOSScoach when joining a new OSS project, I would be able to contribute more *quickly*. |
| U2. | Using a portal like FLOSScoach would improve the *performance* of newcomers contributing to OSS projects |
| U3. | Using a portal like FLOSScoach enable newcomers to increase their *productivity*. |
| U4. | Using a portal like FLOSScoach would enhance newcomers' *effectiveness* on contributing. |
| U5. | Using a portal like FLOSScoach would *make it easier* for newcomers to contribute to OSS projects |
| U6. | I would find a portal like FLOSScoach *useful* to contribute to an OSS project |
| **Perceived Ease of Use (PEOU)** | |
| E1. | *Learning* to operate FLOSScoach would be easy for me |
| E2. | I would find it easy to get FLOSScoach to *do what I want it to do*, to support the first steps of newcomers who want to place their first contribution to an OSS project |
| E3. | My interaction with FLOSScoach would be *clear and understandable*. |
| E4. | It would be *easy to become skillful* in using FLOSScoach |
| E5. | It is *easy to remember* how to perform tasks using FLOSScoach |
| E6. | I would find FLOSScoach *easy to use* |
| **Self-prediction of Future Use (SPFU)** | |
| S1. | Assuming FLOSScoach would be available for any project, I *predict that I will use it in the future* |
| S2. | I would *prefer* using FLOSScoach to the project pages for guiding me contribute to an OSS project |

In the following sections, we present each iteration's results for FLOSScoach's perceived usefulness, ease of use, and prediction of future use. The results are based on the answers collected using the TAM-based assessment instrument presented in Table 4.7.

### 4.4.1 Results for Iteration 1

To assess usefulness, ease of use, and future use, we followed the Technology Acceptance Model method described in Section 4.3.1. In this section, we report on the analysis results of the data gathered from the participants who used FLOSScoach portal during their assignment.

#### *4.4.1.1 Reliability and Validity*

Before presenting the results' analysis, we must check the questionnaire items' reliability and factorial validity. As presented in Section 4.3.1, this should be done to verify whether in this context the instrument and the observations provided reliable and valid results. Even with a small observation set, we checked the assessment instrument's reliability and factorial validity. Surprisingly, we had satisfactory results.

Reliability can be seen as a measure' degree of accuracy within an empirical study. A Cronbach's Alpha reliability level that exceeds a threshold level of 0.8 indicates a reliable measure (Carmines and Zeller, 1979). Thus, both aspects of the adapted TAM can be considered reliable (see Table 4.8).

Table 4.8. Cronbach's alpha for TAM (Iteration 1).

| | Cronbach's alpha |
|---|---|
| **Usefulness** | 0.859 |
| **Ease of Use** | 0.936 |

Factor analysis calculates several factors derived to represent the variation of multi-dimensional data, like our questionnaire questions (Ui and Ei). Table 4.9 shows the adapted TAM questions' factor loading. The literature reports the threshold level for sufficient loading (Laitenberger and Dreyer, 1998) as 0.7; the results for *ease of use* questions E1 to E6 load well with the first factor, thus we interpret this factor as "Ease of Use."

Table 4.9. Factor analysis results (Iteration 1).

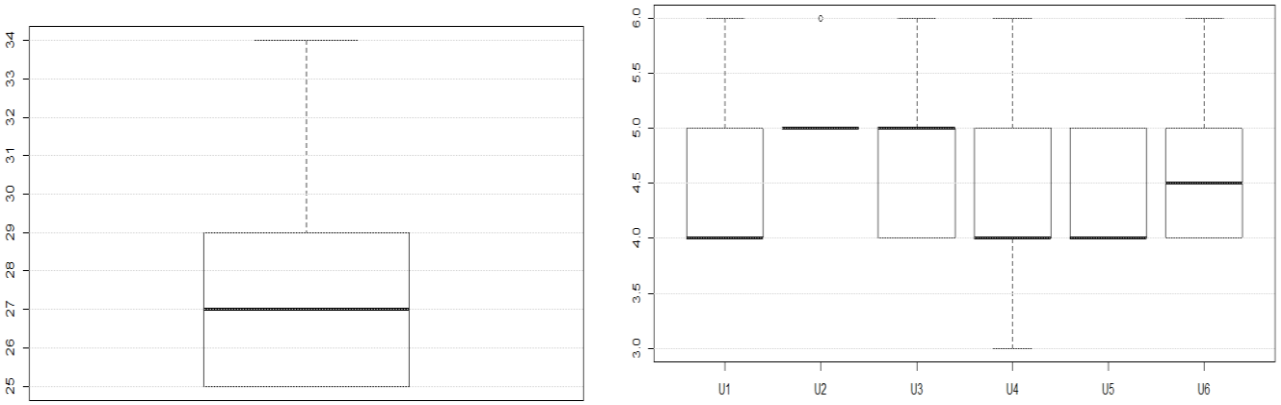
| | Ease of Use | Usefulness |
|---|---|---|
| **Easy to learn (E1)** | 0.52 | -0.82 |
| **Easy to perform (E2)** | **0.97** | 0.22 |
| **Clear and understandable (E3)** | **0.92** | 0.11 |
| **Easy to become skillful (E4)** | **0.94** | -0.14 |
| **Easy to remember (E5)** | **0.90** | 0.34 |
| **Easy to use (E6)** | **0.95** | -0.30 |
| **Work more quickly (U1)** | -0.07 | **0.74** |
| **Improve performance (U2)** | -0.09 | **0.96** |
| **Increase productivity (U3)** | -0.25 | **0.89** |
| **Effectiveness (U4)** | 0.51 | **0.83** |
| **Makes job easier (U5)** | **0.72** | *-0.48* |
| **Useful (U6)** | 0.27 | **0.93** |

We expected the second factor would be loaded by the results of U1 to U6, representing "Usefulness." However, U5 loaded the first factor, and negatively correlated with the second factor. Moreover, in both cases we expected to find a positive loading for all loadings, since, according to reasoned action theory, ease of use and usefulness are correlated factors.

The item U5, which presented high loading in the "Ease of Use" factor, represents subjects' perception that FLOSScoach *made it easier* for newcomers to contribute to OSS projects. This may indicate participants' confusion regarding the item's content. Even presenting satisfactory results, we believe that the small sample (6 observations) influenced the results obtained for reliability and validity. We conducted same analysis in Iteration 2 with a larger sample (18 observations) and present it in Section 4.4.2.

#### 4.4.1.2 *Perceived Usefulness*

Figure 4.12 exhibits the perceived usefulness results. The summative results' ratings, presented in Figure 4.12(a), range between 25 and 34. Considering that the maximum rating is 36, we can conclude that most participants found FLOSScoach useful. We also present each item's results in Figure 4.12(b), wherein we can see that all the medians equaled or exceeded 4. In Figure 4.13, we can see that in most cases (19 out of 36 answers) the participants agreed or strongly agreed with the items. Significantly, all participants agreed or strongly agreed with U2, which concerned the OSS project newcomers' performance during contributing. Amongst all the Usefulness items, the only disagreement noticed was reported for item U4. One participant *slightly disagreed* that FLOSScoach would enhance newcomers' effectiveness in contributing.




(a) Summative usefulness results

(b) Results of usefulness items

Figure 4.12. Boxplots presenting Perceived Usefulness results (Iteration 1).

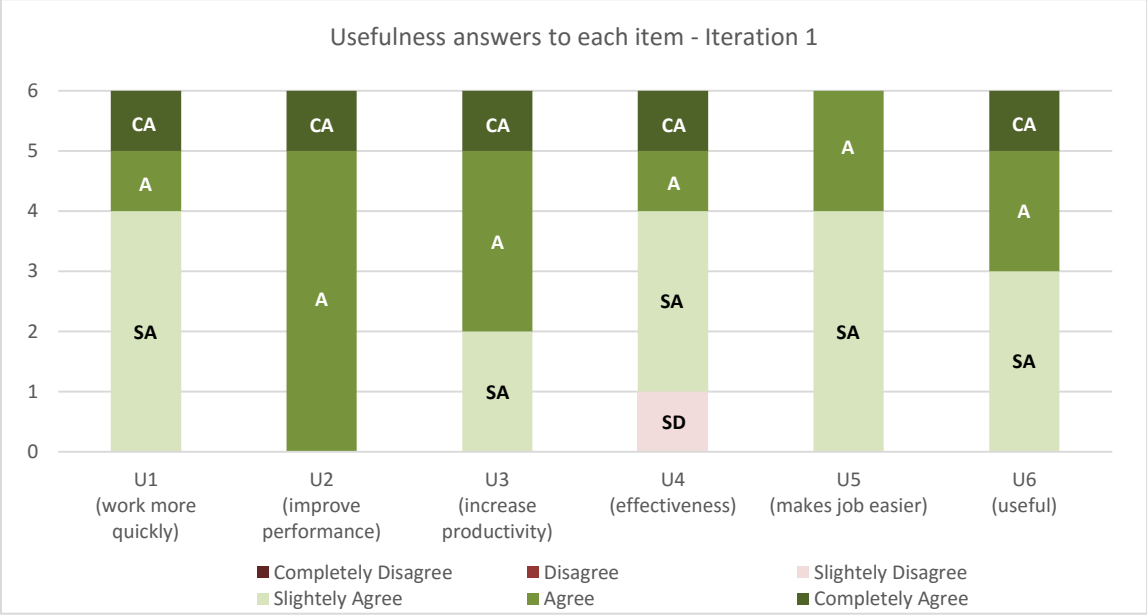
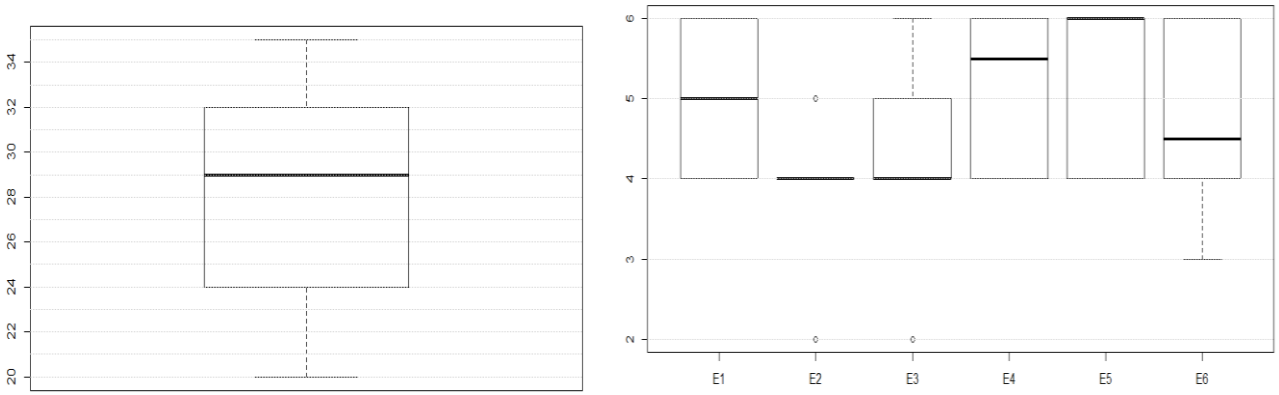



Figure 4.13. Summary of the answers regarding Perceived Usefulness (Iteration 1).

#### 4.4.1.3 *Perceived Ease of Use*

Figure 4.14 and presents the ease of use results. The summative score of the items related to perceived ease of use ranges between 20 and 35. Regarding the maximum rating of 36, we can consider that the portal is overall easy to use. The results are very similar to perceived usefulness. We can conclude that most of our participants found FLOSScoach easy to use.




(a) Summative ease of use results

(b) Results of ease of use items

Figure 4.14. Boxplots presenting Perceived Ease of Use results (Iteration 1).

In Figure 4.15, we can see that the great majority of participants at least *sligthly agree* with the ease of use items. Disagreements were found for items E2, E3, and E6. Analyzing the results we noticed that one participant disagreed with these three items (FC4_UTF); in her diaries we found possible explanations. She was the one who suggested changing the contents' organization and reported the some items' duplicated entries under documentation and technical hurdles. During the debrief session we gathered more information, and she reported: "*I had some issues to find what I was trying to find. There is repeated information.*" We assign the low ratings given to these issues.

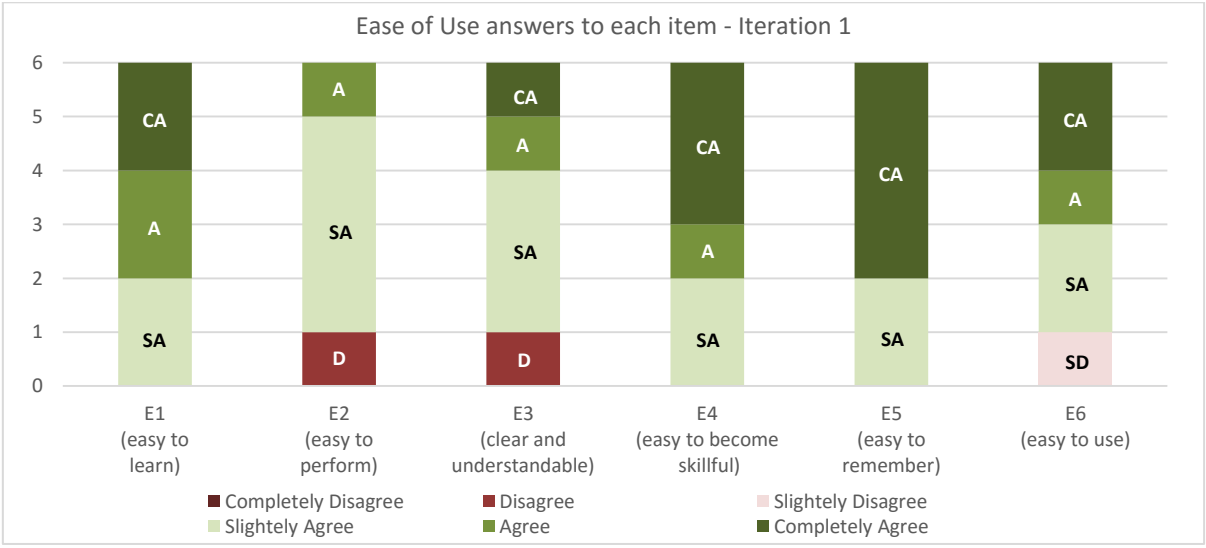



Figure 4.15. Summary of the answers regarding Perceived Ease of Use (Iteration 1).

#### 4.4.1.4 *Self-Predicted Future Use*

Figure 4.16 reports the results of the questions related to self-predicted future FLOSScoach use. We observed that five participants *slightly agree* and one participant *agree* that, if available for a future project, they would use FLOSScoach (S1). We observed the same pattern regarding participants' preference in comparison with the traditional way of finding project information (using the project page).

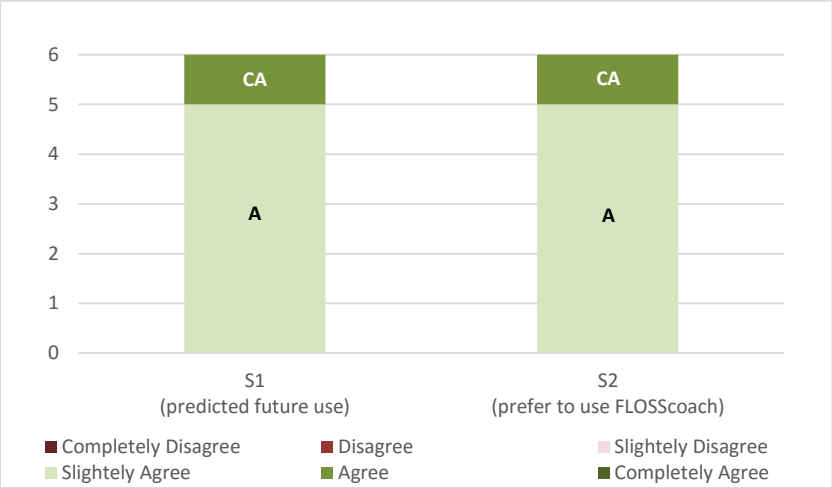



Figure 4.16. Summary for the answers regarding Self-Predicted future use of FLOSScoach (Iteration 1).

In general, we conclude that there is a real chance that the participants will use FLOSScoach if they contribute to another OSS project.

### 4.4.2 Results for Iteration 2

As in Iteration 1, before presenting the analysis of the results, we checked the assessment instrument's reliability and factorial validity.

To verify reliability we calculated the Cronbach's Alpha. Thus, we can consider both adapted TAM aspects reliable (see Table 4.10).

Table 4.10. Cronbach's alpha for TAM's constructs (Iteration 2).

| | Cronbach's alpha |
|---|---|
| **Usefulness** | 0.930 |
| **Ease of Use** | 0.922 |

Regarding Factorial Validity, we present the adapted TAM questions' factor loading in Table 4.11. Since the results for *ease of use* questions E1 to E6 load well with the first factor, we interpret this factor as "Ease of Use."

For this iteration, the second factor loaded well for the results of questions U1 to U6, as expected. We interpret this factor as "Usefulness." We can observe that U6 loads below the

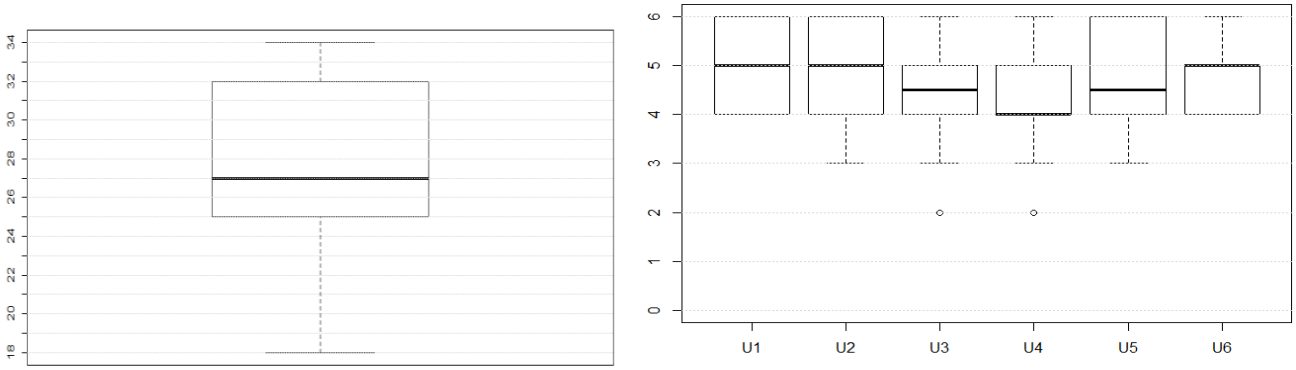
threshold of 0.7. However, since it loaded (slightly) higher on usefulness than on ease of use, we attributed it to the former, as in some other reports (King and He, 2006; Babar et al., 2007).

Table 4.11. Factor validity for TAM's constructs (Iteration 2).

| | Ease of Use | Usefulness |
|---|---|---|
| **Easy to learn (E1)** | **0.92** | -0.23 |
| **Easy to perform (E2)** | **0.70** | 0.40 |
| **Clear and understandable (E3)** | **0.71** | 0.21 |
| **Easy to become skillful (E4)** | **0.85** | 0.16 |
| **Easy to remember (E5)** | **0.87** | 0.23 |
| **Easy to use (E6)** | **0.91** | 0.23 |
| **Work more quickly (U1)** | 0.53 | **0.75** |
| **Improve performance (U2)** | -0.02 | **0.91** |
| **Increase productivity (U3)** | 0.06 | **0.94** |
| **Effectiveness (U4)** | 0.14 | **0.95** |
| **Makes job easier (U5)** | 0.49 | **0.73** |
| **Useful (U6)** | 0.50 | 0.58 |

#### *4.4.2.1 Usefulness of FLOSScoach*

Figure 4.17 exhibits the perceived usefulness results. The ratings of the summative results range between 18 and 34 (median 27.50), as we can observe in Figure 4.17(a). Considering that the maximum possible rating is 36, we can conclude that most participants found FLOSScoach useful.




(a) Summative usefulness results (b) Results of usefulness items

Figure 4.17. Boxplots presenting Perceived Usefulness results (Iteration 2).

To better explore the results, we present each item's detailed results in Figure 4.17(b) and Figure 4.18. In Figure 4.17(b), we see that all the medians equaled or exceeded 4, representing a positive result overall. Figure 4.18 more clearly displays the high amount of participants who agreed with the tool's usefulness. Furthermore, the number of participants who did not agree with U2, U3, U4, and U5 was at most three (for U3 and U4), explaining the whiskers presented in Figure 4.17. We can also observe in the Figures that U1 (quickness) and U2 (job performance) presented median values equal to 5, and that no one disagreed with U1 and U5.

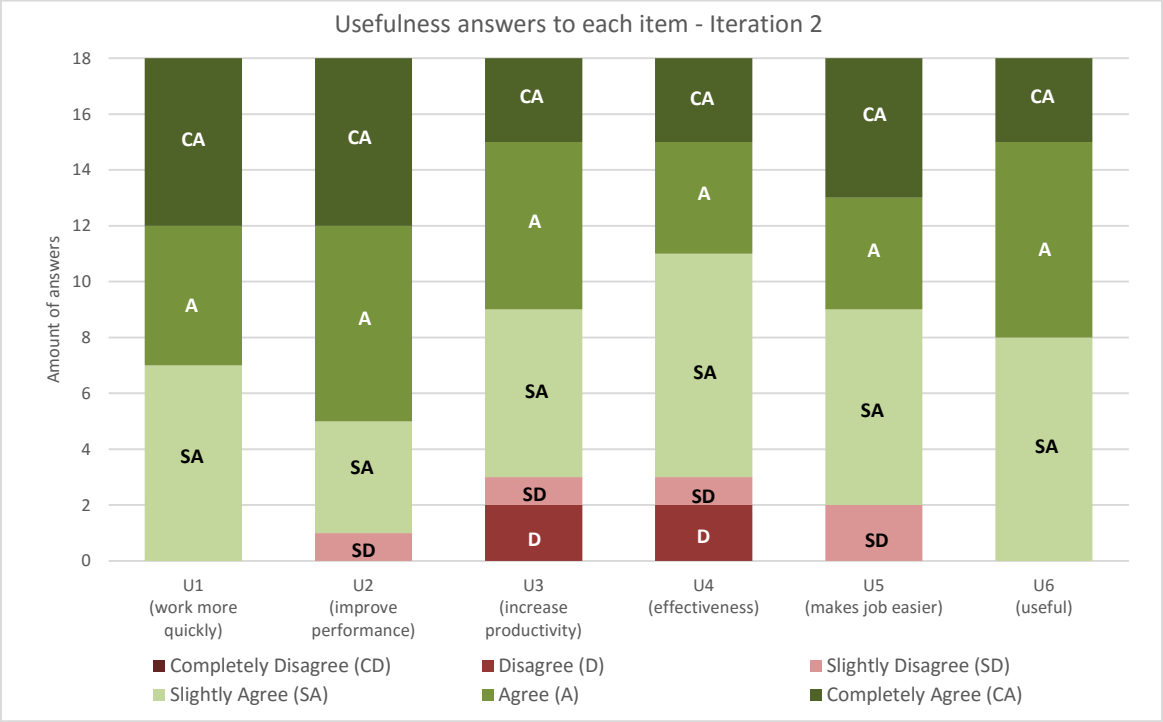



Figure 4.18. Summary of the answers regarding Perceived Usefulness (Iteration 2).

We also present the summative results per subject in Figure 4.19. Observing this figure, we notice only 3 scores below 24 points (which represents mean score of 4 per question): participant FC17_USP, FC15_USP, and FC03_USP. By analyzing the individual rates, we found that three participants provided all 9 negative ratings (disagree, weakly disagree).

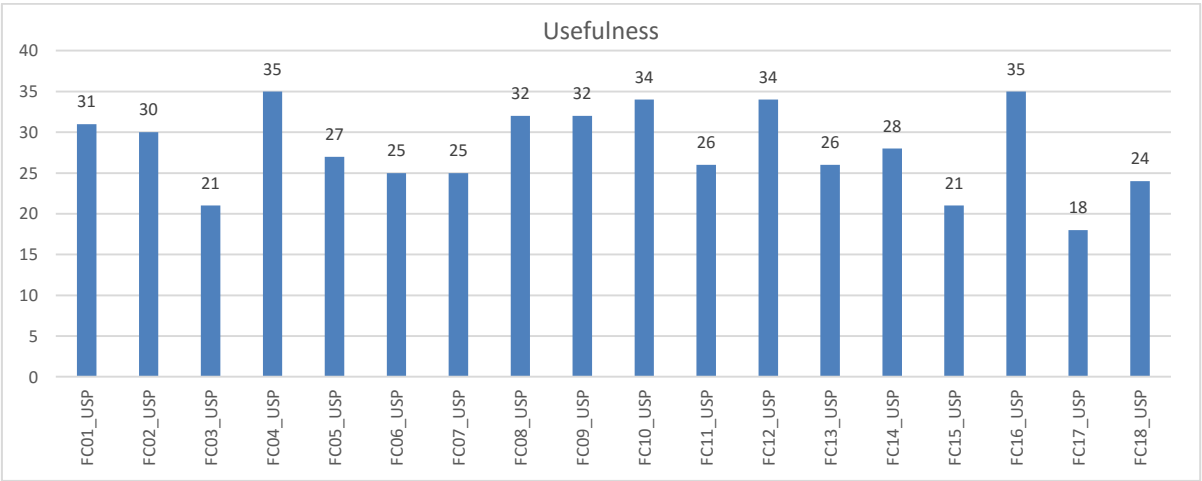



Figure 4.19. Summative perceived usefulness score per subject (Iteration 2).

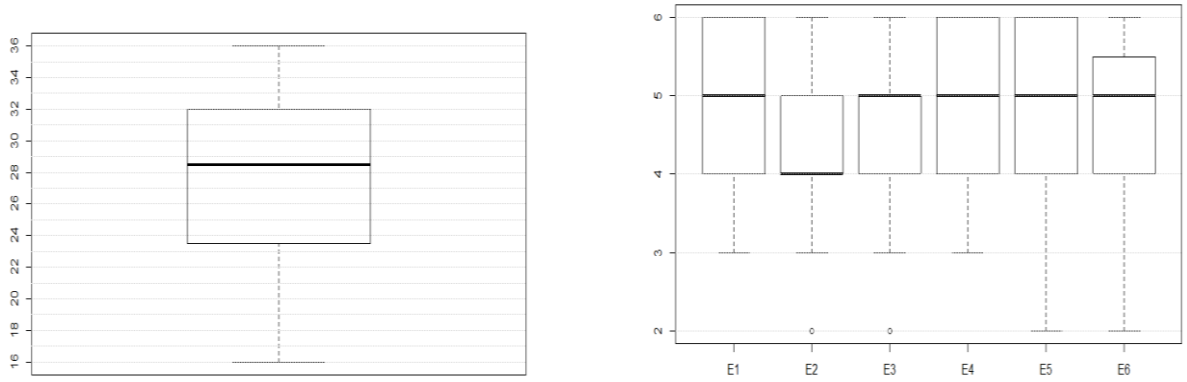
FC17_USP provided the 2 outlier rates observed in Figure 4.17 (Boxplot). From FC17_USP, we discovered an issue with information regarding MacOS building instructions, which were unavailable. Other than that, in the post-study questionnaire he reported that FLOSScoach was "*an excellent starting point, but does not substitute community wiki and discussion list.*" We

understood that, in this case, the participant found FLOSScoach's usefulness limited compared to the projects' wiki. We could not find any explicit reason for FC03_USP and FC15_USP's low scores. For FC15_USP, we can note that, when asked about FLOSScoach's weaknesses or potential areas of improvement, his answer was simple: "*Nothing.*" The same applies to FC03_USP, who answered: "*Nothing in the portal seems unnecessary or bad.*"

#### *4.4.2.2 Ease of Use of FLOSScoach*

Figure 4.20 presents the ease of use results. The questionnaire items' summative score results, which are presented in Figure 4.20(a), range between 16 and 36 (median 28.5). Regarding the maximum rating of 36, again it is possible to say that the portal was considered easy to use. The results are very similar to perceived usefulness. We can conclude that most participants found FLOSScoach easy to use.

In Figure 4.20(b) and Figure 4.21, we can observe the answers' distribution per item. It is possible to observe that more than 50% of the participants agreed or strongly agreed with the items E1, E3, E4, E5, and E6, confirming how easy it was to use the portal. We would like to highlight item E3, showing that 66.67% of the participants agreed or strongly agreed that the portal is clearly structured and understandable.




(a) Summative perceived ease of use results (b) Results of perceived ease of use items

Figure 4.20. Boxplots presenting Perceived Ease of Use results – Iteration 2.

In Figure 4.22, we also provide each participant's summative results. We notice that only 3 participants (FC11_USP, FC13_USP, and FC15_USP) provided all negative rates. We would like to highlight participant FC13_USP, who disagreed or slightly disagreed with all ease of use items. We attempted to find a reason for this behavior in his diaries and post-study questionnaire, and we could not evidence any issue or complaint that justified such low scoring. We contacted the participant, and he answered that he did not remember the reasons, but that it was likely because he would like to find information on how to solve workspace setup issues, which the portal did not offer. We acknowledge that this is lacking, and expect to develop tools for helping newcomers with technical hurdles.

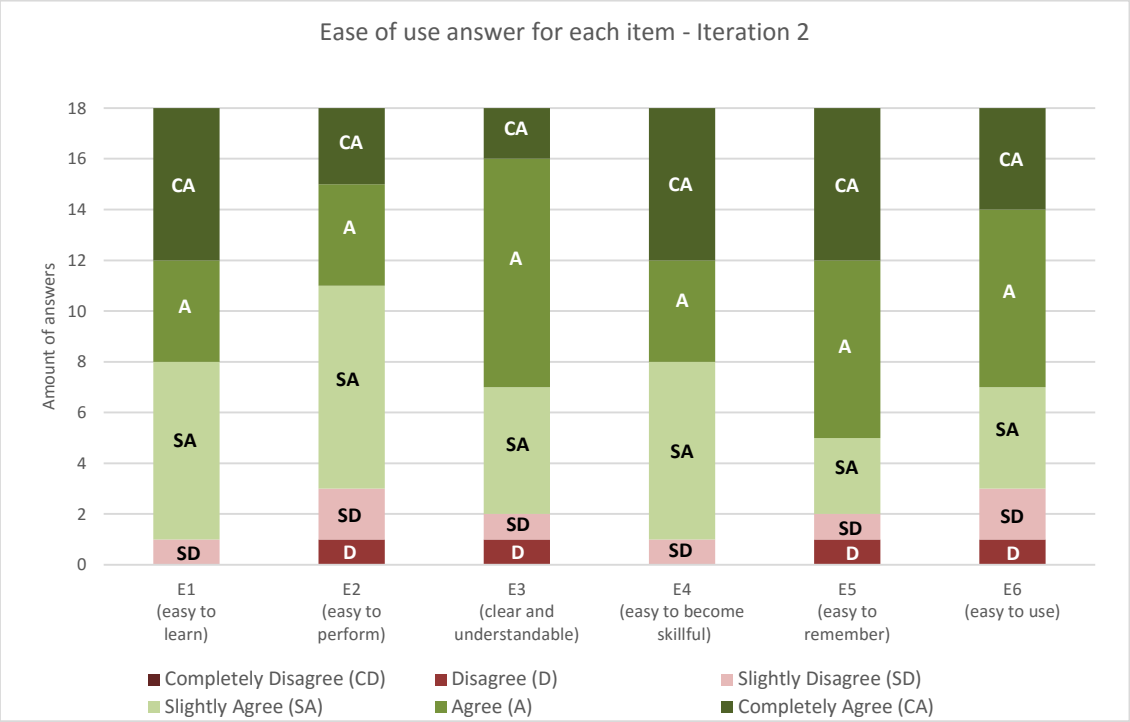



Figure 4.21. Summary of the answers regarding Perceived Ease of Use (Iteration 2).

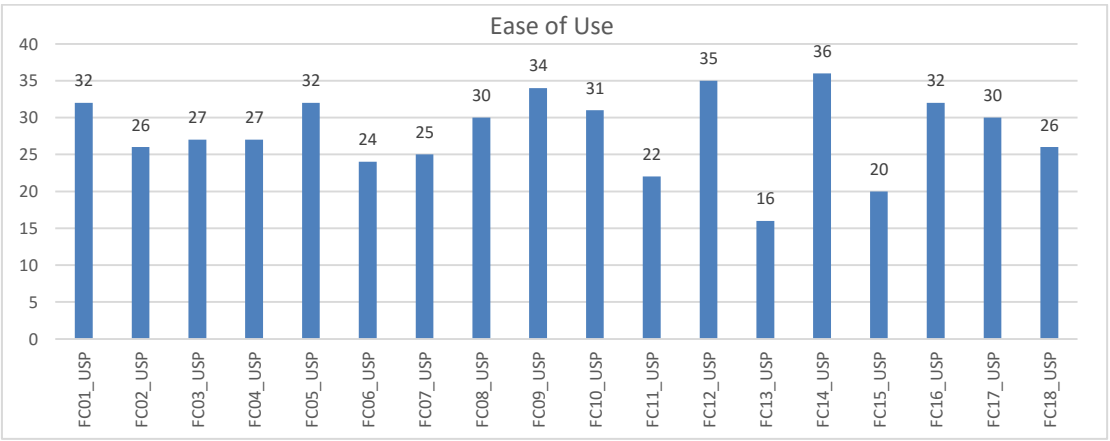



Figure 4.22. Summative perceived ease of use score per subject (Iteration 2).

#### *4.4.2.3 Self-predicted future use*

Figure 4.23 reports self-predicted future FLOSScoach use. We can observe that 14 participants (77.78%) agree or strongly agree that if FLOSScoach were available for any project in the future, they would use it. Compared to the traditional way of finding projects' information (using the project page), a higher number of participants slightly agreed (7 participants) with a preference for FLOSScoach, and three disagreed at some level. FC15_USP appeared again, informing us that he strongly disagreed, i.e., he preferred the traditional form to FLOSScoach, but offered no further explanation.

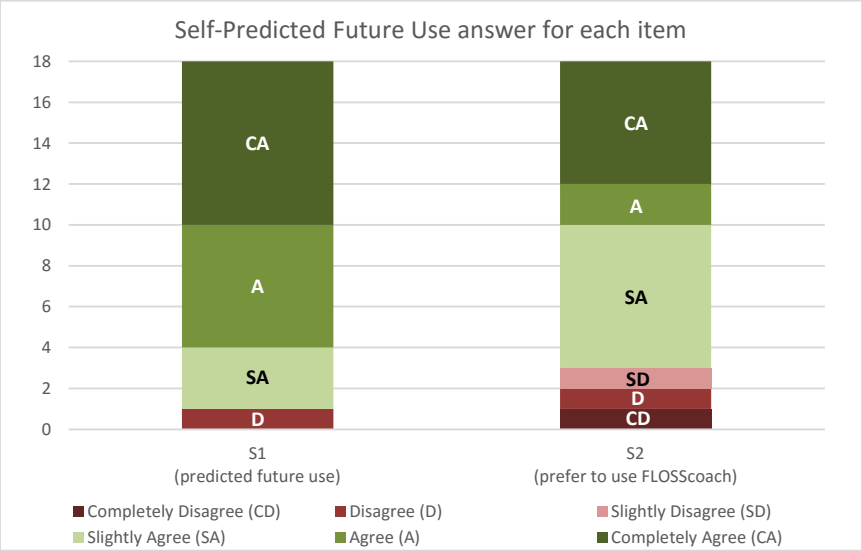



Figure 4.23. Summary of the answers regarding Self-Predicted Future Use (Iteration 2).

#### *4.4.2.4 Correlating Usefulness, Ease of Use, and Self-predicted Future Usage*

Individually analyzing usefulness, ease of use, and self-predicted future use, we did not analyze how participants' opinion influenced user acceptance. We conducted a regression analysis to investigate the correlation of usefulness, ease of use, and self-predicted future usage. According to the TAM model, usefulness and ease of use strongly correlate to self-predicted future use, the intention to use the technological innovation. Table 4.12. presents the correlation among the three factors. We can see that the three factors positively correlate, and, in this case, Ease of Use is an important driver predicting future use of the portal.

These results show that usefulness and ease of use are important indicators for self-predicted future usage. We observed a relation between both variables and self-predicted future usage.

Table 4.12. Correlation among Usefulness, Ease of Use, and Self-predicted Future Use.

| | PU | PEOU | SPFU |
|---|---|---|---|
| **Perceived Usefulness (PU)** | 1.000 | 0.648 | 0.585 |
| **Perceived Ease of Use (PEOU)** | 0.648 | 1.000 | 0.802 |
| **Self-predicted Future Use (SPFU)** | 0.586 | 0.802 | 1.000 |

Although statistical analysis conducted does not imply causality, these results showed that usefulness and ease of use were important determinants that influence and explain self-predicted future use. Most of our subjects consider FLOSScoach useful and easy to use. Moreover, they indicated the possibility of future use, which firmly aligns with what we discovered in the diaries analyzed beforehand.

### 4.4.3 Summary

We can conclude that participants perceived FLOSScoach as a useful and easy to use portal, and would potentially use this tool to support future project contributions. Our results were positive in both iterations, with few negative scores.

## 4.5 Updating the barriers model

Our goal to develop and evaluate a portal to support OSS project newcomers' contributions was intended to verify how the proposed barriers model categories would in practice be used. After developing and assessing FLOSScoach, we found some categories needed rearranging. In this section, we present the new model that was generated after accommodating the observations made during the portal conception, and after receiving the student study participants' feedback.

Therefore, the final model presented reflects the final structure observed in FLOSScoach following the two iterations. We highlight the following changes:

- A category called **communication** was created to accommodate problems related to **newcomers' communication** and community **reception issues**.
- **Cultural differences** barriers are now included in **communication** and **newcomers' behavior** categories. This change was made in response to the barriers' proximity with communication and newcomers' behavior categories, which we perceived during the conception of the portal.
- The **technical hurdles** category was dismissed, making **code/architecture hurdles**, **change request hurdles** and **local environment setup hurdles** first-level categories. This rearrangement was suggested after iteration 1, because the two-level structure for accessing such important information confused newcomers.
- **Documentation problems** that crosscut other categories were accommodated into other categories, which retained 'documentation' in their name. This change was based on suggestions from iteration 1 participants, who reported existing repetition.
- A new barrier **lack of information on required disk space** was added to the **local environment setup hurdles** category.

We present the post-rearrangement model in Figure 4.24.

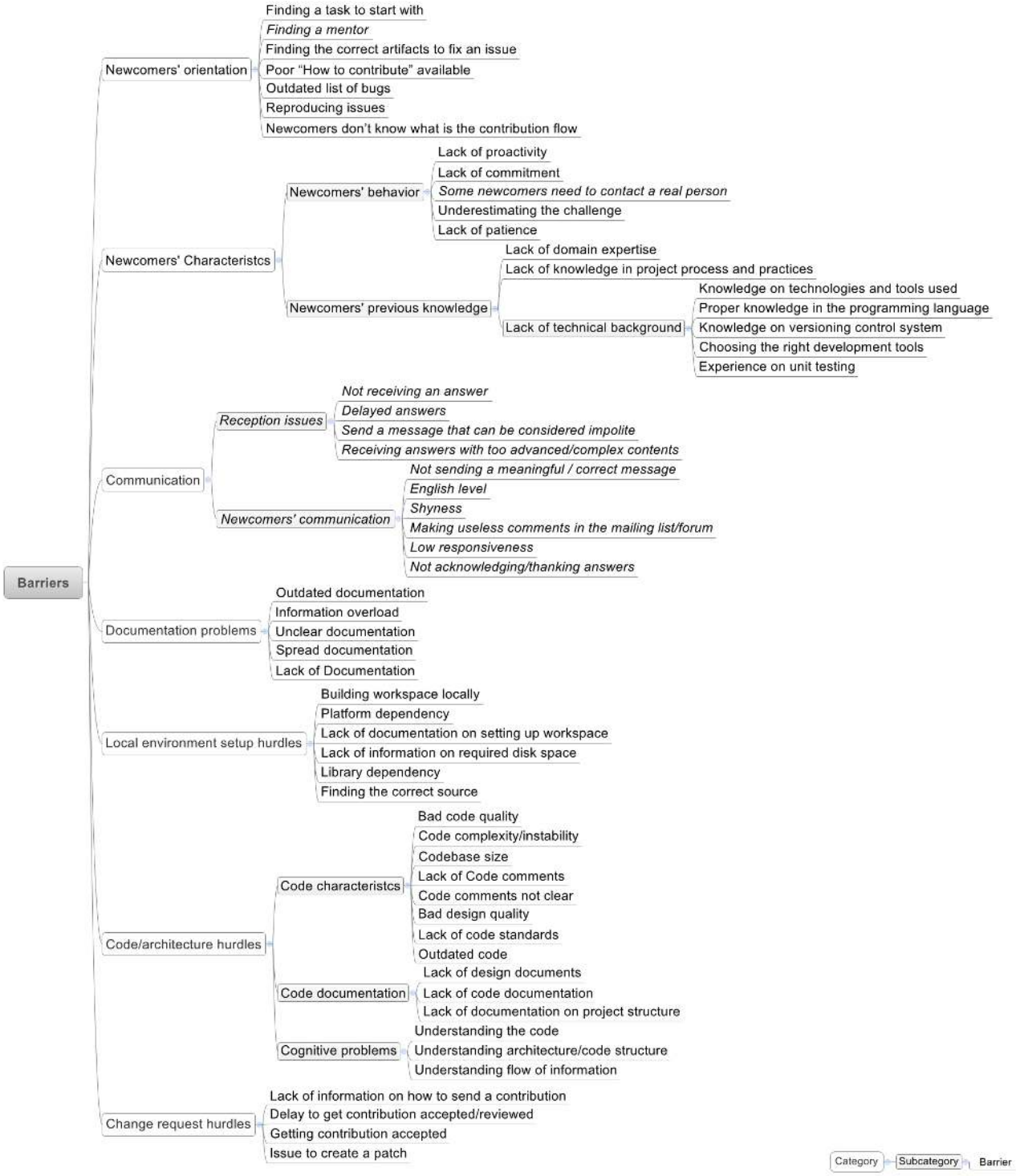



Figure 4.24. Model redesigned after adjustments.

## 4.6 Threats to validity

Although diary studies have high external validity (in that they reflect participants' real-life experiences), there are also drawbacks. First, it is almost impossible to ensure that participants write their diary entries in an unfiltered way. Second, daily diary studies follow a case-study approach. In the diary study we aimed for in-depth understanding, rather than statistical validity.

Regarding the assignment's execution, we acknowledge that, as colleagues, the subjects could talk to each other, and therefore information could leak to the control group. We tried to prevent this by clearly presenting the study's goals and the division of the groups to the students, as well as by explicitly explaining that the evaluation would be based on their efforts and the diary entries. Regarding the students' evaluation and the possible benefits received by the students that used FLOSScoach, we emphasize that we evaluated them according to the process and their efforts. After finishing their evaluation, we conducted a Mann-Whitney to compare the grades of students of control and case groups. The result indicated that there was no statistically significant differences between their grades (p=.3419).

Students can omit information when writing the diaries, especially if they think their grade will be negatively affected by discussing any problems they encountered. We tried to make it clear to the students that lack of previous knowledge or mistakes would not influence their evaluation. Moreover, we read their entries consistently and frequently requested additional information or explanation, which should have counteracted any withholding.

The Iteration 1 participants received training on Open Source Software basics and on writing diaries. We found no implication of any lack of OSS training. Concerning the diaries, we observed that participants of Iteration 2 wrote poorer diaries than the entries provided during Iteration 1, providing, in general, fewer details. This might have affected the results of Iteration 2, since we had less data to analyze and draw our conclusions from. Aiming to reduce this threat, we consistently read and requested additional information or explanations about their entries. However, the students did not respond to our requests with adequate enough information to overcome the threat.

Although we used data from a variety of projects, the findings are not generalizable to all projects, nor can we provide full explanations for the lowered barriers. It is likely that there are problems with the use of FLOSScoach that we did not uncover here; and there are likely other barriers that can be lowered when different individuals use the portal. We are aware that each project has its singularities and that the OSS universe is huge, meaning the level of support and the barriers can differ according to the project. Our strategy of considering different projects aimed to explore different ways to use the portal and overcome barriers.

Another threat to external validity is our use of students, which affects the results' generalization. Moreover, most of our subjects were novices to software development in real settings (with no previous industry experience), and thus it is possible that some barriers they faced are not specific to OSS development. However, as mentioned before, Höst et al. (2000) found that students

perform the same as professionals on small tasks of judgment. Moreover, students are potential contributors to OSS projects.

Additionally, though, students may have felt that they needed to provide positive feedback on the surveys (especially evaluating FLOSScoach). Although we emphasized to both groups that the task had no bearing on their grade, it may still have been uncomfortable for students to criticize work that was known to relate to faculty research.

Another threat to the results' validity is the data subjectivity of the qualitative diary analysis. We grounded our findings in the data collected and presented excerpts to mitigate this threat.

We acknowledge that there are external factors that influence the TAM model, which can affect the results. Moreover, TAM measures were based on self-reported questionnaire items as opposed to objectively measured ones. However, our results showed that the questionnaire items were reliable and valid.

## 4.7 Final Considerations

In this chapter, we presented the application, evaluation, and evolution of our barriers model to newcomer OSS project contribution, which we made by building and assessing a portal to support newcomers. We found that the portal improved newcomers' experiences of the contribution process, serving as a compass during the contribution journey. This finding was evidenced both in the qualitative analysis of their contribution diaries and in the self-efficacy results, which increased for most of the participants who used the portal.

Newcomers who used the portal felt more confident deciding which steps they needed to take to achieve their goals. The subjects highlighted the contribution (step-by-step) flow and the helpful organization of information as FLOSScoach's main beneficial features. However, we could not evidence any significant improvement in supporting newcomers to overcome technical barriers. The study's practical application of the barriers model led to observations that resulted in a new version of the model, which included some rearrangements.

By applying the barriers model in practice, we could evaluate and evolve it to reflect the needs identified in the study's feedbacks. Therefore, we created an updated version of the model with seven categories accommodating 58 barriers.

To triangulate our findings, we gathered feedback from four experienced members we interviewed beforehand, and from two staff members of OpenHatch, whom we considered domain experts. We asked experienced members from JabRef (2 people) and LibreOffice (2 people), since these two projects were included in FLOSScoach's initial version. We received only two answers, one from a LibreOffice member and one from an OpenHatch staff member.

The LibreOffice member summarized his impressions: "*I really liked it. Congratulations, I found it really good!*" He mentioned, "*the portal helps people telling them how to proceed with the*

*chosen project, like how to start, build the software, how to find a task, and many other things that are important to newcomers to any open source project.*" Moreover, he highlighted: "*the most interesting thing is the overview of the project in one panel. Telling the developers what are the programming languages of the project and how to find a task in an easy way.*"

We received more details from the OpenHatch member. She highlighted specific features of FLOSScoach: "*I really like how you worked with projects to add a lot of detail about their setup process, communication tools, etc. ... I really like that in your project overview you list the soft skills required by a project as well as the technical skills.*"

When asked about the benefits for newcomers, she was clear and objective:

> *"I definitely think that it can [support newcomers]. Any time people take the effort to help make the implicit knowledge of open source contribution explicit, that's helpful."*

The small amount of feedback received was very positive. Thus, in consonance with our study's results, the experienced members found the portal useful, indicating that it can help OSS project newcomers contribute.

In addition to the research question findings, we checked results against the influence of previous experience and gender. We only analyzed those participants who were not dismissed. We found that, out of six people who reported having more than 3 years of experience in the industry, five (83.33%) delivered a contribution to the projects (2 had theirs accepted before the end of the assignment). The other subject (FC13_USP) started changing the code, but did not finish fixing the chosen bug. For the other participants who reported having less than 3 years of previous industry experience, we found no difference from the participants without previous experience. From the diaries, in general, we could notice that even the more experienced developers reported technical problems in setting up the workspace and understanding the code. The main difference was that their previous experience helped them to approach and overcome the barriers they faced.

Regarding previous OSS experience, three participants that reported having previous experience completed the assignment, and only FC02_USP submitted the fix (and got it accepted). We found no difference among the barriers faced by these participants and the others. The same conclusion applies to the self-determined level of knowledge in programming languages.

Gender differences were also observed as a possible factor that would influence the types of barriers and behavior. We observed that the self-efficacy for five female participants increased, whereas only two decreased. A significant difference appeared in the diary writing style. Female subjects provided richer entries, expressing their feelings and detailing each problem, solution, and step toward contribution.

In the next chapter, we discuss some findings and results of this thesis, and provide some guidelines for and impacts of this research in practice.

# Chapter 5
# Discussion

According to the Open Source Initiative, "open source" refers to software that can be freely used, changed, and shared (in modified or unmodified form) by anyone. OSS projects are virtual communities of developers who work on a shared set of source code artifacts in order to build specific software products. These communities generally comprise geographically dispersed developers who collaborate via the Internet. These software products' source codes are generally distributed under a license approved by the Free Software Foundation[61] or the Open Source Initiative.[62]

OSS' supposed advantages derive from the fact that source code is freely shared. According to Raymond (1999), the higher number of users and developers involved in the OSS process leads to code quality improvement. Feedback is constant; thus, the software is more robust because many people work on finding, reporting, and fixing bugs. Raymond summarizes with what he called "Linux Law": "given enough eyeballs, all bugs are shallow."

Particularly for this thesis, the most relevant advantage of a shared, public codebase is that it enables any developer who wants to contribute can obtain the code, work on it, and submit changes. Raymond (1999) compares this development model to a noisy bazaar, where visitors can look, touch, taste, add and remove what they want.

In community-based OSS projects, the bazaar structure relies on volunteers' efforts; a project's growth and sustainability requires a continual influx of newcomers. Newcomers maintain the number of "eyes" necessary to correct software defects and add additional product features (Capiluppi and Michlmayr, 2007). Therefore, a steady influx of newcomers often correlates to an OSS project's success (Bird et al., 2007; Capiluppi and Michlmayr, 2007; Qureshi and Fang, 2011; Sinha et al., 2011).

Some aspects of newcomers joining of OSS projects can be negative. For instance, since new developers are less productive when they first join a new project, average project productivity

[61] http://www.fsf.org

[62] http://opensource.org

decreases. Moreover, newcomers joining a project may require more coordination efforts, since a project with more people requires more coordination (Cockburn, 2000). In addition, communication efforts could also increase, since the amount of messages exchanged among the team increases. These threats revolve around the two main premises that underlie Brooks' Law: productivity (the ramp up problem) and communication/coordination overhead (Brooks, 1995).

The Brooks' Law (Brooks, 1995) states that "adding manpower to a late project makes it later." At first glance, the statement does not seem to apply to community-based OSS projects because they rarely follow a strict release schedule; it would be difficult to determine if a project was "late." However, it is possible to interpret the meaning of "late" from sponsors', users', and participants' viewpoints.

In the context of this thesis, we do not consider the ramp up to reach high productivity as a threat. We understand that not all newcomers want to become core developers (or productive developers), and that some newcomers contribute not to join a project, but rather only to make few, sparse contributions to it. This scenario is similarly theorized by the crowd development approach (LaToza et al., 2013), in which the development process is idealized as a set of micro tasks implemented by different distributed developers.

We also believe that by providing adequate support to newcomers, coordination overhead can be reduced. Besides, Eric Raymond, in his well-known paper "The Cathedral and Bazaar" (Raymond, 1999), claims that the effects of Brooks' Law do not apply to communication/ coordination overhead, since the more peripheral developers work on separable parallel subtasks and interact with each other very little. Therefore, according to Raymond, coordination overhead has its effects within the core group. This assumption is supported by other studies found in the literature (Johnson, 2001; Koch, 2004; Schweik et al., 2008; Capiluppi and Adams, 2009).

When it comes to newcomers' joining process, adherence to Brooks' Law is one of the differences between traditional commercial software projects and OSS projects. One difference regards newcomers' motivation. Whereas in commercial settings newcomers join a project as part of their duties when hired, OSS newcomers have a wide range of motivations, including some developers who only want to make one contribution. Moreover, joining a commercial project usually involves formal training sessions on technical and organizational topics, as well as mentorship programs, in which a more senior developer is assigned to support the newcomer.

Notably, although we considered the barriers as a negative aspect of onboarding, some barriers can be used as filters. Findings from some studies (Ducheneaut, 2005; Halfaker et al., 2013a) revealed that certain entry barriers led to improved future contributions. Moreover, research conducted in the OSS domain demonstrated that socialization barriers beneficially maintain community integration and the quality of the community's product. Since our focus was on newcomers making their first contribution, regardless of their intention to become a community member, we were not concerned about overall socialization issues or future contributions, thus distinguishing our study from the existing literature.

Regarding the literature, we found that previous research focused on socialization problems, appearing in 75% of the studies analyzed during our systematic literature review, with a heavy focus on mailing list interactions (receiving response and socialization with other members). We also noticed a lack of in-depth studies on technical issues faced by newcomers, which may be due to the small number of qualitative studies found, because it cannot be quantitatively extracted from mailing lists. For example, **code/architectural hurdles** are evidenced by only five studies analyzed, and should be further explored. Issues related **to local environment setup hurdles** were reported in only one study by one subject in a debrief session. These issues deserve more attention from both practitioners and researchers.

Regarding social barriers, some are well reported and analyzed by the literature; **reception issues** are the most explored. **Not receiving an answer** is quantitatively well evidenced in both OSS and Question and Answers (Q&A) literature. **Receiving impolite answers** was also largely studied in CSCW, mainly through the analysis of Wikipedia reverts. The proposed strategies of automated answers and feedback used in Wikipedia can be adapted and then evaluated in the OSS context. On the other hand, some barriers identified by our study are neglected in the literature. One of them is **receiving answers with too advanced/complex contents**. We could not identify any study dealing with this barrier. There are opportunities to investigate this kind of barrier in different domains, and thereby provide a deeper understanding of this issue.

Overall, we presented, substantiated, and characterized the barriers OSS project newcomers face, which OSS researchers can use to develop newcomer support strategies. By providing the context of CSCW and related research literature (Section 3.2.6) and developing FLOSScoach, we provided a starting point to design such support. To fully develop newcomer support, more attention must be paid to specific research topics, such as understanding (and creating ways to measure) the influence of the barriers on newcomers' experience, and identifying and creating different strategies to lower each barrier. We also believe that communities from other CSCW domains can benefit from our research. Future research ought to focus on searching for commonalities and differences amongst barriers faced in different domains in order to build models and theories about open collaboration communities' joining processes.

OSS practitioners can adapt the existing strategies listed in the literature and use FLOSScoach to provide their own newcomer support. Assuming that, as stated by Dagenais et al. (2010) "*newcomers are explorers who must orient themselves within an unfamiliar landscape*," the barriers model and FLOSScoach portal can serve as the starting point for open source communities to support newcomers, alerting them about the barriers that they might face. The positive results we obtained from using our model to build a newcomer portal should encourage communities to invest in this direction, and persuade researchers to continue investigating additional support strategies. In addition, in the following section we present a set of guidelines that can be followed by OSS communities and newcomers.

## 5.1 Initial set of guidelines

Based on our interviews and our observation of students attempting to contribute to OSS projects, we drafted a set of guidelines useful for projects that want to offer appropriate newcomer support, and newcomers who want to contribute to OSS projects. We did not evaluate the guidelines; however, we consider it an important outcome for both OSS communities and newcomers willing to contribute.

### 5.1.1 Guidelines for OSS projects

**Answer quickly**. Remember that participants are people who chose to spend time trying to help. Do not let their motivation decrease. *Do not make the newcomers wait or leave them without answer. Automatic greetings* could help (Preece, 2004), at least to send the message that someone will answer them quickly, or to guide them to another possible communication channel.

**Be kind and make newcomers feel part of the team**. Make newcomers feel welcome; treat all of them as potential contributors and show them that the community cares about them. Experienced members should answer newcomers' messages and welcome them, even if they have already received an answer. *Designating a few experienced members* to deal with new members, or setting a code of conduct can potentially solve reception issues. Send thankful, welcoming messages to account for cultural differences and misunderstandings.

**Local/regional communication channels**. Whenever possible, create regional mailing lists, IRC channels, and forums. This type of resource can encourage newcomers' first contact and reduce the barriers related to cultural difference and language.

**Create a newcomer-specific page**. Give the newcomers every resource they need, and only the resources they need. Do not flood newcomers with every possible resource, since too much information can confuse them. Show only what is important for newcomers' first steps, like how the project is organized, and what/where are the important resources (code repository, mailing lists, issue tracker, IRC channel, code review tool). Keep the page clean, organized, up-to-date, and easy to follow. Make this space a kind of "new developers' guidelines" section.

**Set expectations and needs early**. Show newcomers what is expected from them, where the difficulties lie, and what skills and level of expertise they need to have (what programming languages and technologies are used by the project, etc.). Place this information somewhere that newcomers access early in their journey.

**Show the path, the easiest path.** Create an easy to access, easy to follow path on which newcomers can start their journey. Leave breadcrumbs, left intentionally for them, to guide the newcomers to easy tasks. The newcomers need to get rewards soon, so the less energy they dedicate to overcoming the initial set of barriers, the better. This is what Fogel (2013) called '*hacktivation*

*energy*'. Projects need to "*bring the hacktivation energy down to a level that encourages people to get involved.*"

**Point newcomers to easy tasks**. If there is no way to map a special path for newcomers, at least tag the issues to help newcomers find suitable tasks for new contributors. Some informative tags that can guide newcomers include: difficulty level, module affected, language/technology skills needed, and members who can help. We present some projects that use tags to guide newcomers in Section 5.2.

**Create different kinds of tutorials and documents.** Provide videos, demos, screenshots, and step-by-step tutorials for the different contribution phases. Make these tutorials simple and clear, especially for more complicated activities.

**Make it easy for newcomers build the system locally.** Setting up the local workspace was the most reported barrier in our study that demotivated and frustrated many newcomers. One option for helping newcomers overcome setup barriers is to create a step-by-step detailed tutorial, which is linked to information about usual problems and possible solutions (an excerpt of FAQ section).

**Keep the issue list clean and triaged**: Read the issue list frequently in order to clean outdated tasks and triage/tag issues. Outdated, comment-less issues can be scary to newcomers.

**Document the processes and practices**: Document all design decisions, processes and practices defined and make them easily accessible. Information like code/naming standards, patch submission process, task assignment process, communication practices, design decisions etc. need to be properly documented, helping newcomers in their first contributions. Remember to present the technical and social processes and practices.

**Dismiss or identify outdated information**: if it is hard to keep documentation up-to-date, community members should remove outdated information or, at least, clearly identify it as outdated. Making newcomers aware of the absence or the status of a document can save their time and set their expectations. By recognizing the absence or obsolescence of some documents, communities can request help from the newcomers to update or create such documentation.

**Create a live FAQ section**. Create FAQ sections to help developers finding answer to recurrent questions. The FAQ must not be a static section; it must be live and grow according to questions and issues recurrently asked or reported. The community can build the FAQ cooperatively, in a wiki-like page, enabling anyone to contribute with entries (question + answer).

**Keep the code as simple as possible**. Code is supposed to be read by humans. Refactor the code constantly to make it more readable. The source code is the set of artifacts that need to be understood and changed by the members to contribute. Sometimes it is not easy to make it simple because the core of a large application is inherently complex, and the code reflects this complexity. However, directing newcomers to peripheral modules (at least warning them of the complexity) would benefit newcomers during their first steps.

**Use and update code comments**. It makes it easier for newcomers to understand the code.

**Document the code structure**. Clearly document the way that the code is organized, and how the components, modules, classes, packages relate to each other. A thorough documentation about the structure and relationship among its modules can make it easier for the new developers to understand the code and find the artifacts they need.

### 5.1.2 Guidelines for newcomers to OSS projects

**Find the project resources**. Search the project page to find the source code repository, issue tracker, and available communication means (mailing lists, forums, IRC channels). This is basic information you will use during your journey.

**Be proactive and try possibilities**. Try to solve the barriers you face by searching for solutions in the mailing list archives and other resources the community provides.

**Do not fear the community**. Think of the community as your safe harbor. If you could not solve your problems by yourself or with the help of the available the resources, make contact with the community. Check what communication means are available, and talk to the community.

**Send a kind and meaningful message**. Here are some hints on how to send a message to the community: be kind; present yourself by mentioning your skills and your goals; clearly and objectively ask your question; share what you tried to do to solve your problem before referring to the community.

**Never join a flaming war**. Sometimes you can receive a rude answer; if so, ignore it and continue looking for what is important to you. If the answer does not address your issue, apologize for the misunderstanding, thank the respondent, and rephrase your question.

**Fight hard to set up the local workspace**. In general, setting up a local workspace is a time-consuming and demotivating activity. If available, follow a tutorial to guide you, and always refer to the community for support.

**Use a virtual machine to set up your workspace**. Problems can arise with previously installed packages, dependencies from other software, and unsupported versions of operating systems. You might also face problems if you use a distribution that differs from the project-recommended one. Installing a new operating system in a virtual machine prevents you from facing some of these problems, and from crashing already-installed applications.

**Find an easy task to start with**. Try to find out if the project tags their issues with specific keywords that identify suitable tasks for newcomers (easy hack, easy task, good for newcomers, etc.).

**Keep the community informed about your decisions**. If you choose a task to work with, send a comment in the issue you are working on informing others that you are trying to address it. In the

same way, inform whenever you give up, or find any problem related to that task. This way, you are contributing to the community and avoiding concurrent work.

**Help the community to support other newcomers**. Whenever you face a barrier and figure out how to solve it, you can help other newcomers overcome it by reporting your problem and solution to the community. Send your report in the mailing list, or add it to the community's FAQ.

## 5.2 How is the state-of-the-practice situated?

There are some OSS project initiatives to support and facilitate newcomers' onboarding. The strategies we report here were identified by lurking projects, following mailing lists, and interviewing core members.

Most of the projects offer "how to contribute" or "introduction to development" pages. These pages present general information such as how to contribute, paths to obtaining the current codebase, mailing list addresses and etiquette code, introduction to the issue tracker system, building guidelines, and so on. In some cases, these resources are too high-level, do not present all the necessary information, or are outdated, as we learned during some interviews.

Some projects provide additional special strategies for new contributors, including the well-used "filtering the issues and bugs," which aims to provide newcomers a list of "easy-enough" tasks. Three large projects make extensive use of this strategy to support newcomers. First, LibreOffice created a section called "Easy Hacks" on their Wiki page. When newcomers enter the development Wiki, they find a link for the Easy Hacks page.[63] In this page, they can find the tasks, suggested newcomers workflow (presented in Figure 5.1) and lists of tasks filtered by difficulty, skills and topics (as presented in Figure 5.2). Apache Open Office also provides this approach, likely because these two projects branched off from the Open Office project, after Oracle acquired Sun Microsystems and they inherited some initiatives.

[63] https://wiki.documentfoundation.org/Development/Easy_Hacks

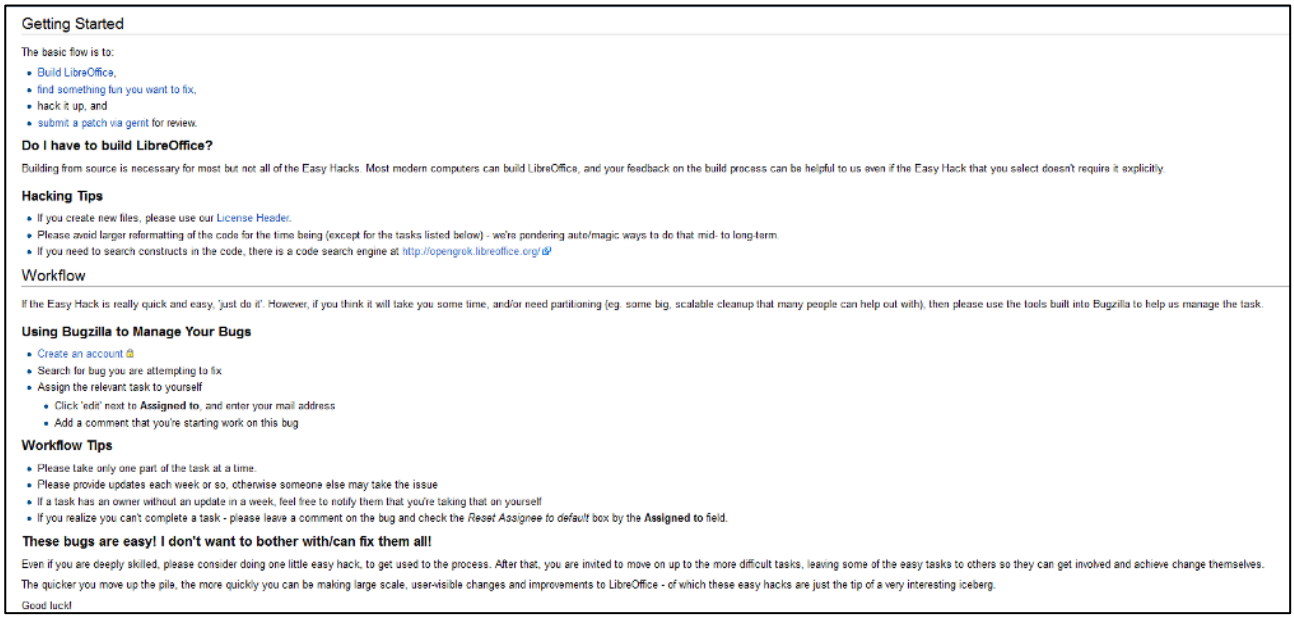



Figure 5.1. "How to Contribute" wiki page of project LibreOffice.

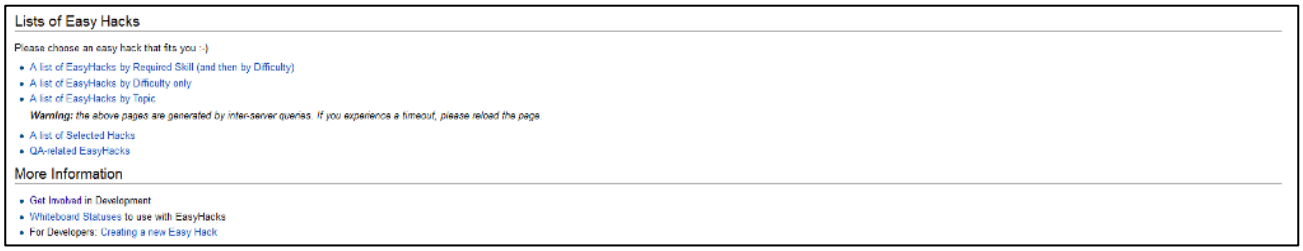



Figure 5.2. List of Easy Tasks for Newcomers to LibreOffice.

Mozilla also offers this kind of strategy, enabling newcomers to find easy, or, as they call them "good" first bugs.[64] In the same way, MediaWiki[65] provides "*a curated list of bugs that are relatively easy for a new MediaWiki developer to fix, or tiny features you could add.*" [66]

In Ubuntu[67] products, new developers can also take advantage of the task-filtering approach. They present a list of "bitesize bugs" that a newcomer can easily solve. Interestingly, this approach is highly stimulated and supported by the community. For example, lurking in the ubuntu-devel[68] mailing list, we found a discussion on the subject "We need more bitesize bugs." From Ubuntu, we also found a great explanation as to why tagging a bug can support new developers:

*"We get more and more people who are excited about helping out, who read some documentation and are just blocked on finding a simple task they can get started with."*

[64] https://developer.mozilla.org/en/docs/Introduction
[65] https://www.mediawiki.org
[66] https://www.mediawiki.org/wiki/Annoying_little_bugs
[67] http://www.ubuntu.com/
[68] https://lists.ubuntu.com/mailman/listinfo/ubuntu-devel

In addition to filtering the bugs, Mozilla and Ubuntu also provide mentoring programs. In these initiatives, some bugs include an assigned mentor. In Mozilla, for example, some bugs are mentored. In Mozilla's page, they are presented as bugs that *"have a mentor who commits to helping you every step of the way. Generally, there should be enough information in the bug to get started. Whenever you need help, contact the mentor over IRC, in the bug itself, or by email. When you've completed the bug, they will help you get your code into the tree."*

Apache[69] also offers a mentoring program that focuses on providing mentors to anyone interested in contributing their effort to an Apache Software Foundation (ASF) project. It is a formal program,[70] whereby newcomers can subscribe to receive a mentor. Analyzing several other projects, we found that in some cases the mentorship is less formal, and occurs via IRC channels where newcomers can locate experienced developers who can support them with technical issues.

There is a more pervasive and well-known mentoring initiative in OSS projects called Google Summer of Code. In this program, Google provides scholarship for students interested in writing code for OSS projects, and the selected projects assign mentors to support the students during their scholarship period.

Additionally, LibreOffice's community organized a wiki page called "Find the Expert" where they list a set of developers who are experts in specific knowledge areas within the project. The list is used to add the appropriate developer(s) to the list of followers of a bug, or to find a reviewer of a proposed change. Having the most appropriate people added to a bug is helpful, as they would pay close attention and clarify any doubts or questions.

Recently (mid-November, 2013), a discussion started in the LibreOffice developers' mailing list about providing a virtual machine mirror with a setup development environment, which would help to avoid initial workspace setup problems. One developer defined the idea as *"such a 'get started right away' environment would be quite valuable... it might make sense to have a moderately recent checkout of master fully built in it, as it likely is not that much extra space and really lets a developer start right away."* Some developers supported this idea, and one of them explained how it could support newcomers:

> *"While doing a* build *might feel encouraging, it's not too rare that newcomers somehow mess up their tree and since they are insecure are not sure if they can fix that in ways an experienced dev confidently does. As such, having a one-click way to get back to that 'just after* build *'-state is certainly very comforting."*

An initiative that crosscuts OSS projects is OpenHatch.[71] One of their goals is lowering open source communities' entry barriers. They provide a list of easy bugs for dozens of projects, try to match newcomers' and projects' skills, and point to possible volunteer mentors who could help

[69] http://apache.org
[70] https://community.apache.org/mentoringprogramme.html
[71] http://www.openhatch.org

newcomers during early stages. Moreover, they promote events at university campuses to help teach community knowledge and technical skills.

In the future, we plan to conduct an in-depth domain analysis as a future project, aiming to map the actual, existing strategies adopted by OSS communities onto our barriers model.

## 5.3 Impact of this research in practice

This project's intermediate results triggered discussions amongst practitioners and in IT specialized media and blogs. This is rewarding and encouraging, since it indicates that our research can offer insights and support to OSS communities. In this section we briefly present some of the citations that our papers received.

A deep discussion on a 'hacker' forum started in response to a link to our SLR (Steinmacher et al., 2014c) on August 1, 2014.[72] This post seemed to trigger subsequent discussions on mailing lists. An email[73] sent to Mozilla Community Building Team mailing list presented the findings of our Systematic Literature Review (Steinmacher et al., 2014c) in order to reinforce the need for proper newcomer reception on Mozilla. The sender mentioned that: "*This study on 'Barriers faced by newcomers to Open Source' was an interesting read ... I think a good strategy can be to ensure someone owns community channels, where it's implicit that they call out behavior, and reinforce what good response looks like*."

Some members from other communities spread the word about our research on developers' mailing lists, discussing what they do right and what else they need to do to better support newcomers. Others simply mentioned our paper with no further discussion.

A member of OpenWorm[74] (OW) sent our paper (Steinmacher et al., 2014c) to the project mailing list[75], and acknowledged the existence of the barriers in the project: "*I think this paper outlines a number of issues that the OW project is experiencing.*"

In a message to firefox-dev mailing list,[76] presented in Figure 5.3, a member reported that they are doing things right, according to the evidence from our SLR paper results (Steinmacher et al., 2014c).

[72] https://news.ycombinator.com/item?id=8121833
[73] https://mail.mozilla.org/pipermail/community-building/2014-August/001788.html
[74] http://www.openworm.org
[75] https://groups.google.com/forum/#!msg/openworm-discuss/5aPM-7ivZ90/9z0sa41bd88J
[76] https://mail.mozilla.org/pipermail/firefox-dev/2014-August/002039.html

**Some community engagement news.**

**mhoye** mhoye at mozilla.com
*Fri Aug 1 10:33:47 PDT 2014*

- Previous message: Possibility of having small FOSS projects in introductory CS c
- Next message: Some community engagement news.
- **Messages sorted by:** [ date ] [ thread ] [ subject ] [ author ]

---

This is partly a repost, as it was suggested I'd find a wider audience for it. A bit of a wall of text, sure, but I think it'll be worth your while. There are bits in here about what we're doing right, what we need more of, and what we can learn to do better.

Here's an interesting paper that came out recently: " Barriers faced by newcomers to open source projects: a systematic review", by Igor Steinmacher.

http://www.academia.edu/6537077/Barriers_faced_by_newcomers_to_open_sour

You might want to skim it, but you'll be happy to hear that:

- We are doing basically everything right, and
- Nothing in there comes as a huge surprise.

That's a pretty good feeling, but it's important to have a feeling like that backed up with evidence.

One nice thing about that paper is that it's got a couple of numbers in it that continue to matter a lot to us. Getting back to new contributors with 48 hours remains super-important. Likewise DXR links in good first bugs and tags that help matchmaking contributors' skills to the bug's needs.

Mentors that are helpful, thank contributors and point people to the next bug they can work on. We do these things because we're all civilized people who care about our work and our community, obviously, but it's nice to know that in addition to feeling good, this stuff is what works.

Figure 5.3. Email sent to firefox-dev mailing list.

Another mention of our paper appeared in the Apache community-dev mailing list.[77] A member mentioned our paper (Steinmacher et al., 2014c) as a resource for guiding communities toward making projects easier for newcomers.

The message mentioned above was forwarded to incubator-blur-dev.[78] In this case, they included two suggestions of actions that their community could take, based on our results:

"*1) intentionally leave some small, approachable issues unfixed for a while (e.g. BLUR-351[1]). Promote them on twitter/mail list.*

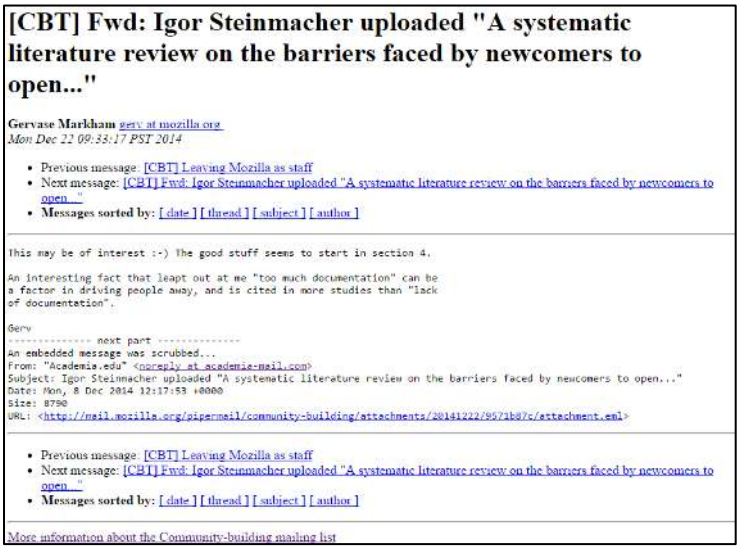
*2) create some docs on the overall structure of the code, how the projects fit together, maybe highlight some 'paths' through the code.*"

In December, another email was sent to the Mozilla Community Building Team mailing list,[79] mentioning our SLR published at Information and Software Technology (Steinmacher et al., 2014c). The message, presented in Figure 5.4, referred to the paper's results sections, and highlighted a finding regarding documentation overabundance as something that drives away new contributors.

[77] https://mail-archives.apache.org/mod_mbox/community-dev/201408.mbox/%3C53E1715A.9010000@gmail.com%3E

[78] http://mail-archives.apache.org/mod_mbox/incubator-blur-dev/201408.mbox/%3CCAB6tTr15pE2nZv7On8KU7szFsauFtdUynO9XOkMHQVCJwLWWBw%40mail.gmail.com%3E

[79] https://mail.mozilla.org/pipermail/community-building/2014-December/002404.html



**[CBT] Fwd: Igor Steinmacher uploaded "A systematic literature review on the barriers faced by newcomers to open..."**

**Gervase Markham** gerv at mozilla org
*Mon Dec 22 09:33:17 PST 2014*

- Previous message: [CBT] Leaving Mozilla as staff
- Next message: [CBT] Fwd: Igor Steinmacher uploaded "A systematic literature review on the barriers faced by newcomers to open..."
- **Messages sorted by:** [ date ] [ thread ] [ subject ] [ author ]

This may be of interest :-) The good stuff seems to start in section 4.

An interesting fact that leapt out at me "too much documentation" can be a factor in driving people away, and is cited in more studies than "lack of documentation".

Gerv

------------ next part ------------
An embedded message was scrubbed...
From: "Academia.edu" <noreply at academia-mail.com>
Subject: Igor Steinmacher uploaded "A systematic literature review on the barriers faced by newcomers to open..."
Date: Mon, 8 Dec 2014 12:17:53 +0000
Size: 8790
URL: <http://mail.mozilla.org/pipermail/community-building/attachments/20141222/9571b87c/attachment.eml>

- Previous message: [CBT] Leaving Mozilla as staff
- Next message: [CBT] Fwd: Igor Steinmacher uploaded "A systematic literature review on the barriers faced by newcomers to open..."
- **Messages sorted by:** [ date ] [ thread ] [ subject ] [ author ]

More information about the Community-building mailing list

Figure 5.4. Email sent to Mozilla Community Building Team mailing list.

Bitcoin developers presented our paper (Steinmacher et al., 2014c) during a meeting[80] to motivate a discussion about newcomers to Bitcoin. They mentioned that the social barriers highlighted in our paper were also present in Bitcoin, and that they need to welcome new developers if they are to build a sustainable community.

Apart from mailing lists and discussions, we also found citations to our research in other relevant media and publications. One reference to our paper (Steinmacher et al., 2014c) appeared in a post[81] from Amy Tsay (community manager at Mozilla) in her weblog MozAmy, talking about good relationship- and community-building at Mozilla.

Matt Asay mentioned our systematic literature review (Steinmacher et al., 2014c) twice. In August 2014, he published an article in InfoWorld[82] commenting on and discussing the findings of our SLR. They discuss it from the practitioners' side, with nice comments and examples. An excerpt of this article is presented in Figure 5.5. Later, in October 2014, he mentioned our study in his blogpost at readwrite.com.[83] Asay used the findings of the paper to talk about the newcomer reception in OSS communities. Figure 5.6 displays the title of the article and the excerpt that mentions our research.

[80] https://www.youtube.com/watch?feature=player_detailpage&v=l7kgqv1zOxo#t=2381

[81] http://mozamy.wordpress.com/?p=71?preview=true&preview_id=71&preview_nonce=225945de00

[82] http://www.infoworld.com/article/2608819/open-source-software/open-source-software-how-to-crack-an-open-source-community.html

[83] http://readwrite.com/2014/10/17/linux-linus-torvalds-community-mistakes-toxic-environment

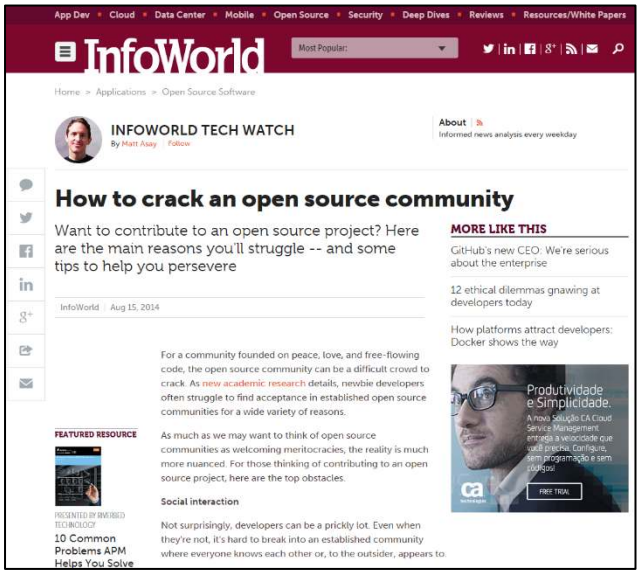



Figure 5.5. Part of the article "How to crack an open source community" published in InfoWorld.

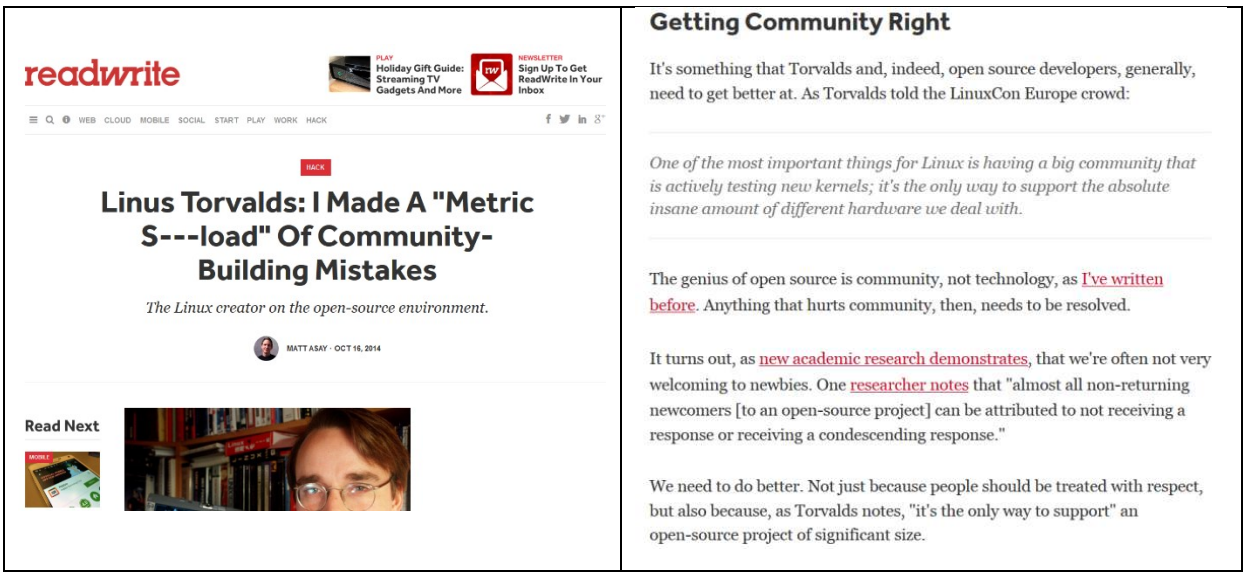



Figure 5.6. Title and part of the article that mention our paper at readwrite.com.

Recently, our paper published by Information and Software Technology (Steinmacher et al., 2015c) was mentioned by Chris Aniszczyk (the head of Open Source at Twitter) in one of his tweets.[84] A screenshot of his post is presented in Figure 5.7.

[84] https://twitter.com/cra/status/539430993724047361


Figure 5.7. Tweet of Chris Aniszczyk about our paper.

One of our papers (Steinmacher et al., 2015a) is referenced in "Mozilla Community-Building Readings"[85], in wiki.mozilla.org. We present an excerpt of the wiki in Figure 5.8.


Figure 5.8. Mozilla Community-Building Readings wiki page mentioning our paper.

Our research also received mentions by OpenHatch community members. We found two IRC logs citing our publications,[86, 87] and an issue[88] reporting that they appreciated the list of soft skills presented in FLOSScoach; they suggested making this information available in OpenHatch as well.

It was very pleasant to see our research's influence in practice. Some of our results were both used and discussed in different projects, and were mentioned by practitioners in different media. This shows that our research had immediate and relevant implications for OSS communities, and offered an opportunity for the communities to discuss and think about newcomers' experiences.

[85] https://wiki.mozilla.org/Contribute/Reading
[86] http://irclogs.jackgrigg.com/text.pl?server=irc.freenode.net;channel=openhatch;date=2014-09-04
[87] http://irclogs.jackgrigg.com/text.pl?server=irc.freenode.net;channel=openhatch;date=2014-11-23
[88] https://github.com/openhatch/oh-mainline/issues/1501

# Chapter 6
# Conclusions

Numerous OSS communities are maintained by volunteers, who can easily drop out. Communities must exert explicit effort if they are to mitigate the barriers that new contributors face (Wang et al., 2012). In this thesis, we identified, organized, and discussed key barriers that hinder newcomers' first contribution to OSS projects, and proposed and evaluated FLOSScoach, a portal to support newcomers.

In Phase I of this thesis, our goal was to identify the barriers faced by OSS newcomers. We relied on data extracted from interview and questionnaire responses from developers in different stages of the OSS community joining process, student feedback, and a systematic literature review (SLR) on OSS newcomers' barriers. The main result of this phase was a preliminary model comprising 58 barriers grouped in six different categories. This model organizes the existing research evidence as well as common knowledge about the barriers faced by OSS project newcomers. Notably, 50 out of 58 barriers presented in the resulting model were identified in the practitioner interviews. We also observed that less than 30% of the barriers (17 barriers) appeared in the literature, which primarily focused on social barriers.

In Phase II, our goal was to apply, evaluate, and evolve the preliminary barriers model, in order to support the first contributions of OSS project newcomers. We built a portal using information gathered from practitioners and evaluated it in a study with undergraduate students, which relied on qualitative data from diaries, self-efficacy questionnaires, and the Technology Acceptance Model (TAM). By analyzing newcomers' diaries, we found that the portal helped newcomers take their first steps. The newcomers who used the portal felt more confident during the contribution process and about the information provided. The portal guided the newcomers like a map or compass, evidencing that organizing the information properly and providing a friendly environment to newcomers can support their first steps. Moreover, the results of TAM evidenced the portal's usefulness and ease-of-use, and the participants indicated their possible future use of the portal.

Another outcome of the barriers model's development and of the diary analysis was the evolution of the model. Among the changes, we added the barrier **lack of information about required disk space** under **local environment setup hurdles**. Many other rearrangements were made to reflect the changes necessitated during conceptual development and suggested by the

participants of the study. The resulting model comprises 58 barriers, organized into 7 main categories.

Quantitative results obtained from questionnaires demonstrated that newcomers' use of FLOSScoach positively influenced their self-efficacy; most increased their self-efficacy score. By analyzing the results per question, we found statistical evidence that portal use made newcomers feel more comfortable and confident about contributing to OSS projects. However, in line with the analysis of the diaries, we did not evidence improvements in regard to technical issues.

We also analyzed the portal's perceived usefulness, ease-of-use, and predicted future use. The results were very encouraging, revealing that newcomers perceived FLOSScoach as a useful and easy-to-use tool to support their first OSS project contribution. The newcomers also indicated a possible future use of the portal. Our results were very positive in both iterations, with more that 83% in agreement on all evaluated items. The items "Using a portal like FLOSScoach when joining a new OSS project, I would be able to contribute more quickly" (U1) and "I would find a portal like FLOSScoach useful to contribute to an OSS project" (U6) were positively evaluated by 100% of the participants.

The artifacts generated in Phase I and Phase II represent this thesis' main contributions. In Phase I, we empirically identified the barriers newcomers to OSS projects face and created a conceptual barriers model. This model offers a starting point for practitioners who want to streamline newcomer onboarding, and lays a foundation for building better support tools. We expected to make communities aware of the problems that can hinder first contributions, thereby offering them an opportunity to think about newcomers. We partially achieved this goal, as it was possible to notice in the discussions and blog posts (see Section 5.3).

In Phase II, we built FLOSScoach, a portal to support OSS projects newcomers, which was based on our barriers model and the Phase I interviews. We evaluated the portal with students and the results are very encouraging. Currently, FLOSScoach is available for public use, prepared to receive newcomers, and ready to prompt further studies and accommodate new tools and strategies for mitigating barriers that hinder newcomers.

We also claim that this thesis contains some minor or secondary contributions:

**Positioning the social barriers in the context of CSCW and related literature**. We observed that the social barriers we identified are fairly similar to those reported in the literature on other open collaboration communities, mainly Wikipedia. Thus, some of the solutions and mitigation strategies used in those contexts could be tried on OSS communities and vice versa. Therefore, we characterized the social barriers faced by newcomers to OSS in the context of CSCW and related literature. Researchers and practitioners can take advantage of this discussion by adapting existing strategies used in other domains to design their own tool-based support.

**The method used to develop and evaluate the portal for newcomers** can be reused in other studies, including evaluating new mechanisms to evaluate additional newcomer barriers. We did not evaluate or discuss our method at length in this thesis, but we consider it a secondary contribution.

**The joining process model** presented in the introduction and used to organize the literature is novel. The newcomers' joining process is often modeled as a distinctly staged process, through which newcomers need to play specific roles until they become a core project member. We were not interested in analyzing the way a newcomer becomes a project or core member, and we did not find a model that described the problem we were dealing with. In order to situate our problem and delimit the scope of this thesis, we analyzed the existing literature and proposed a model that represents the common stages (onboarding and contributing) and influential forces drawing newcomers in or pushing them away from a project.

**Initial guidelines for newcomers and projects** is also a minor contribution that can be used in practice by communities that want to provide a smoother onboarding to the newcomers; and also by newcomers, to guide them during their first steps in an OSS project.

## 6.1 Future work

**Model to measure how newcomer-friendly a project is**. A possible extension for this study is to conduct a series of studies to understand and define ways to measure each barrier's influence on newcomers' experiences. This metrics' definition can be initiated by a broad domain analysis, aiming to catalog the information, strategies, tools, and mechanisms used by OSS projects. These metrics can be used to determine the level of support offered by projects, and then assign projects a grade according to it.

**Mining software repositories to confirm the barriers**. Some of the identified barriers can be further analyzed using software-mining techniques. A possible future direction could be to use these techniques to verify which barriers a given project presents. This kind of study can be used as one of the indicators of a project's friendliness.

**Another look at the barriers**. The barriers can be evaluated from the perspective of 3C collaboration model (Fuks et al., 2007), enabling a different perspective on the findings.

**Barriers faced by other kinds of newcomers**. It would be interesting to conduct the same method to identify the barriers faced by newcomers that begin contributing to an OSS project in non-coding activities including translation, bug triaging, bug reporting and user support.

**Women in OSS.** Investigate the barriers that are related to OSS newcomers' gender in order to foster diversity in OSS.

**Migration to social coding environments**. Recently, many projects are migrating to social coding environments (e.g. github, bitbucket). Researchers could analyze how this migration influenced the newcomers' contributions; for example, by comparing the number of newcomers'

contributions; exploring the barriers that were mitigated by such environments and the role of the social environment; and/or exploring possible new barriers introduced.

**Are barriers always bad?** A clear future work is to explore in more depth how communities perceive barriers and how they affect the quality of newcomers' contributions. Analyzing which barriers are considered 'natural filters' and which ones need to be lowered or eliminated can help achieve a better understanding of the contribution process and of newcomer onboarding.

**User Experience Assessment of FLOSScoach**. Apply user experience assessment methods to evaluate how newcomers perceive and use FLOSScoach.

**Adding collaborative features to FLOSScoach.** A future direction related to FLOSScoach is making the portal a collaborative environment, where newcomers can interact with the portal by editing the pages, rating information, and gaining reputation for their posts and for help provided to other newcomers.

**Automatically feeding FLOSScoach**. An interesting future research direction could be to use mining techniques and natural language processing to add and update project information to FLOSScoach.

# Appendix A
# Systematic Literature Review Protocol

This protocol refers to the conduction of a systematic literature review (SLR) based upon guidelines established for the Software Engineering domain (Kitchenham, 2004; Kitchenham and Charters, 2007; Kitchenham and Brereton, 2013). In this protocol we specify the research question and its components, establishing the requirements regarding sources and primary studies selection, the evidences to collect, and the method of synthesis of such evidences.

## A.1. Research Questions

Newcomers present different technical skills, time availability, and reasons to join to an OSS project. Notwithstanding, 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. Our main goal is to identify the barriers faced by newcomers to place their contributions. By identifying those barriers, it is possible to adapt and create tools and methods to ease the joining process that benefit both newcomers and the project itself. We expect that, by reducing the barriers, projects can benefit from more occasional contributions and more long-term contributors.

Thus, we defined the following as our main research question:

- **RQ 1.** What are the barriers that hinder the newcomers contribution in OSS projects?

By answering this question, we aimed to capture barriers a newcomer face when contributing in an OSS project. We are not interested in newcomers' motivation to contribute to a project, but in the issues they can face after deciding to contribute to a project. We also make no distinction regarding the size, quantity, or frequency of contributions made by newcomers.

A common approach regarding the establishment of the search expression is its characterization based on PICO components: population, intervention, control, and outcome. Based upon previous studies, we knew most papers on the subject comprises of case studies and employs

quantitative, qualitative, or mixed method. Thus, establishing a control element was not feasible. It is worth noticing that recent guidelines on systematic literature review on Software Engineering also orient to omit further characterization (Kitchenham and Brereton, 2013). Based upon these recommendations, we have kept the population and intervention components, and omitted comparison and outcome.

- **Population**: Open Source Software
- **Intervention**: Newcomers contribution

Analyzing these attributes, we can derive the following keywords and synonyms to be used afterwards, when searching for studies:

| Keyword | Synonym |
|---|---|
| Open Source Software | OSS; open source; free software; FLOSS; FOSS |
| Newcomer | Newbie; new developer; new member; new contributor; new people; new committer; novice; beginner; potential participant; joiner |
| Contribution | Contributing; joining process; first steps, entry; initial steps; onboarding; retention; entrance |

## A.2. Search Expression

Using the terms identified in Section A.1, the following (generic) search expression was defined:

```
(("OSS" OR "Open Source" OR "Free Software" OR FLOSS OR FOSS)

AND

("newbie" OR "new developer" OR "new member" OR "new contributor" OR "new people" OR "new committer" OR "novice" OR "beginner" OR "potential participant" OR "joiner" OR "contributing" OR "joining process" OR "first steps, entry" OR "initial steps" OR "onboarding" OR "retention" OR "entrance"))
```

The expression need to be refined to get a reasonable number of results. To evaluate the query we will verify the titles of the papers and also check if a set of already know studies (Krogh et al., 2003; Cubranic et al., 2005; Ducheneaut, 2005; Jensen et al., 2011; He et al., 2012) are retrieved.

## A.3. Study selection criteria and process

The search process will encompass two approaches. The first one was based on queries in digital libraries. However, as reported in the literature, using just this mechanism, especially for systematic reviews in software engineering, is often inefficient. As suggested by others (Jalali and

Wohlin, 2012; Kitchenham and Brereton, 2013), we will apply a single step citation analysis (snowballing) to complement the search process.

Regardless of the mechanism used to search, we will screen the studies according to some criteria pertinent to the research question. We established the following criteria for inclusion of studies.

I1. Full papers must be available

I2. The papers must be written in English

I3. The studies must explicitly mention that they deal with newcomers/joining process in open source software projects

I4. Studies must present experimental results

I5. Only papers published in journals or workshop and conference proceedings will be considered

The exclusion criteria will consist on discarding:

E1. Duplicated papers

E2. Studies that, clearly, deal with topics not relevant to the purpose of this review

E3. Studies regarding newcomers, but not in Open Source Software

E4. Studies about open source software that do not study problems/obstacles faced by newcomers

E5. Studies whose full paper are not available over the internet and also would not be obtained contacting the authors

E6. Paper that presents a previous version of a more complete study about the same topic

E7. Documents that are TOC, event/tutorial/panel/workshop/conference descriptions, book chapters, thesis, editorials, prefaces, news, comments and reviews

E8. Early access or unpublished studies

The selection will comprise the following steps:

(i) Document search: the search will be conducted by using the search string to query the selected sources. The references of the retrieved studies will be stored in a local repository to be further analyzed.

(ii) Title, abstract and keywords analysis: the preliminary selection with the search string does not guarantee that the studies retrieved meet the inclusion criteria and answer the research questions. In this sense, titles, abstracts and keywords will be read to verify which studies meet inclusion/exclusion criteria. The studies will be read by more than one researcher. The researchers must come to a consensus when the evaluations are

conflicting. In case there is no consensus or doubt, the study must be included to avoid premature exclusion.

(iii) Author snowballing: we will also check other paper published by the authors of selected studies, to verify if their work was extended and new publications are made available. To find other publications we will access authors profiles in the libraries used and also check their DBLP profile and personal homepages (when available).

(iv) Introduction and conclusion analysis: the studies must be evaluated regarding their objectives and results. This analysis enables the researchers to further verify if the papers answer the research questions and meet the inclusion/exclusion criteria.

(v) Backward snowballing: studies found and selected by the search on the digital libraries and by author snowballing will be analyzed regarding their references. We will submit candidate papers to the same process used for papers found on digital libraries and author snowballing: screening by title, abstract, and keywords; analysis of introduction and conclusion.

## A.4. Data extraction

After the definition of the primary studies list, the documents will be read by the researchers. The data extracted must support the answer to the research questions. Specifically, we are interested in empirical evidence of barriers that influence newcomers from placing their contributions to Open Source Software projects. We will create a list of barriers evidenced by each paper. Each barrier will be linked to information about the study and type of evidence that were used. After this identification, each barrier will be also linked to text segments that supported it in the papers they were identified. Using the text segments, we will classify the barriers applying an approach inspired on procedures of Grounded Theory (GT).

To enable a more complete review of the studies, the following data will also be extracted:

- Which projects are the study objects?
- What kind of study was conducted?
- What data was analyzed?
- Which data support the factor identification?
- What are the characteristics and goals of the newcomers studied?

## A.5. Sources selection

The criteria used to select the digital libraries to be used to query the studies are:

- It must index papers on open source software research (preferably the main research journals and events on open source software and software engineering).
- It must index papers written on English.
- It must support searching using Boolean expression (at least the operator OR and AND must be supported).
- It must provide access to the complete text of the paper (preferably in PDF format).

In addition to digital libraries, we will also check other paper published by the authors of selected studies.

The Digital Libraries selected for this systematic review were:

- ACM
- IEEE
- Scopus
- Springer Link

We will also send emails to well known researchers in Open Source Software domain in order to request their opinion regarding the forums that we must query to gather relevant studies. The selected researchers are:

- Imed Hammouda, Walt Scacchi, Giancarlo Succi: program chairs of the International Conference on Open Source Systems of 2012 and 2013
- Gregorio Robles: program chair of OpenSym 2013
- Carlos Denner: program chair of WSL 2013

# Appendix B
# Interview documents

In this appendix we present the email used to recruit the participants of the interview and the interview guides used for experienced members, newcomers that succeeded/onboarding newcomers and dropouts.

## A.6. Recruitment email

The email sent to the mailing lists and potential answerers was based on the following:

```
Hello,

my name is Igor Steinmacher and I am a Research Scholar at University of
California, Irvine. I am conducting a research aiming at finding how to support
new contributors during their first steps in the project. My final goal is to
verify what kind of tooling is appropriate to support the newcomers overcoming
their difficulties when they are willing to contribute to the project.

The first step of my research is to find out what are the main obstacles and
difficulties faced by these newcomers. My goal is to hear from the community
itself, interviewing new contributors and core members.

By having the information regarding the main problems, I will start figuring out
which mechanisms can be applied to provide the support. I will keep the
anonymity of interviewers and I have the commitment to publish/return the
results of my research to the community.

I need your help answering my interview. We will conduct the interviews via
textual chat, and we can schedule it at the time that fits better for you.
Please send me a private email if you are interested in supporting my research.

I also request your help reaching out contributors so that I can gather the best
information. The data will help gain insights about newcomer issues, and allow
us to propose initiatives to alleviate the problems faced by newcomers, as a
tentative to retain them.

To volunteer you must be 18 years or older and have experience in software
development.

Feel free to contact me in case you have any doubt or question regarding my
research or the interview process.
```

## A.7. Interview guides

All the interviews followed a semi-structured script and were conducted using textual based chat tools, like Google Talk. We chose this mean once the participants are used to this kind of tool for their professional and personal activities. The interviews were conducted following three different scripts, used according to the participant's profiles. The scripts were validated during the pilot interviews and by one specialist in qualitative studies, and one specialist in Open Source Software.

Following, we present scripts used per profile:

**[Script 1] Experienced Members:**

PROFILE
- For how long are you contributing to Open Source Software projects?
- Is this the first Open Source project you contribute to?
- How many years of experience do you have as a software developer?
- When did you start in this project?
- What is your current role in the project?
- When you started, what was your background in Open Source Software?
- are you paid to develop for the Open Source project?
- is it your main occupation?
- how old are you?
- how much time do you spend contributing to Open Source?

[ ] Less than 5 hours/week
[ ] From 5 to 10 hours/week
[ ] From 10 to 20 hours/week
[ ] More than 20 hours/week

- what is/was (are/were) your main motivation(s) to contribute to Open Source?

**About newcomers**
- What do you think about newcomers in the project?
- What is the profile expected? (EXEMPLIFY: technical skills, tools that need to know, commitment, types of issues he/she is expected to address, …)
- Is there a way to identify a potential newcomer that will keep contributing?
- Is there any kind of special attention policy to receive newcomers in the project?
  (Is there any different approach when member note there is a newcomer?)

**Joining process**
- How do newcomers generally start their interaction? (mailing list, direct emails, patches, issue tracker, reporting bugs…)
  - What do you consider most appropriate (what do you recommend)?
  - How do newcomers can be aware of this?
  - What are the steps that you consider important for a newcomer to start his/her contribution?
  - How can the newcomers be aware of these steps?
  - Do you remember any case of newcomer that succeeded in the project? What about your case?
  - Can you tell what difficulties you (or the case you are reporting) faced in the beginning?

- What about cases of newcomers that gave up, do you remember? (Do you know why ?)

**Problems faced**

- In your opinion: what are the main problems that a newcomer can face when joining the project? (Hints: environment setup, patch process, find the right issue, find the piece of code that you need to touch, who are the right guys to talk to…)
- What are the activities in which newcomers require more support? (setup, coding, standards, finding issues, documentation)'
- Is it easy to get support from the community? Are newcomers answered and guided on their first interactions?

**Existent solutions**

- Is there any kind of effort to alleviate the problems a newcomer face when joining a project?
- Are there any mechanisms that can facilitate some steps of newcomers joining process?
    - Can you think about any kind of tooling or dashboard that could be implemented/designed to offer support?

**[Script 2] Succeded newcomers and Onboarding newcomers**

**PROFILING**

- When have you started in the project?
    - For how long have you been contributing to Open Source Software projects?
    - Is this the first Open Source project you contribute to?
- How many year of experience do you have as a software developer?
- What is your role in the project?
- What was your OSS knowledge when you joined?
    - are you paid to develop for the Open Source project?
    - is it your main occupation?
    - how old are you?
    - how much time do you spend contributing to Open Source?
    [ ] Less than 5 hours/week
    [ ] From 5 to 10 hours/week
    [ ] From 10 to 20 hours/week
    [ ] More than 20 hours/week

**Motivation**

- What motivated you to start contributing to an OSS project?
- Why specifically this project?
    - Was the community receptive?
    - Why do you keep contributing?

**JOINING**

- How have you started contributing? Using which mean?
    - How/why did you choose starting this way?
    - Did you know how to behave and what were the initial steps?

- Which tasks had you conducted when you started?
    - How did you choose these tasks?
        - Were these tasks appropriate? (Do you consider?)
        - Have you gave up or changed the first task for any reason? Why?
    - Did you know how to start executing the task? How have you found the way?

```
            - What kind of information you needed to find to conduct the task?
            - Was it hard to find the information?
- And currently, what tasks do you perform?

Problems
- How were your first interactions with other project members (were they
receptive and helpful?)
- What are your main "frustrations" with the project (considering joining
process)?   - How did you handle those?
       (Examples:       |asked for help? --> To who? Where?
      `           |---> lurked the documentation?
                  |---> tried yourself? )

- How have you coordinated your tasks with other members?

Suggestions
- What steps do you consider important for a newcomer to follow to become a
contributor?
      - Were these steps clearly presented to you?
      - How had you learnt?

- Which mechanisms would you propose to reduce the problems faced by newcomers
when joining the project? Can you think about something?
```

```
[Script 3] Dropout newcomers


PROFILING
- Do you have experience in Open Source Software?
      - For how long had you being contributing to OSS?
- Do you have experience in the language(s)/framework(s) used in the project?
- Are you user of the software?
    - For how long have you contributed to this project?
    - are you paid to develop for the Open Source project?
    - is it your main occupation?
    - how old are you?
    - how much time do you spend contributing to Open Source?
    [  ] Less than 5 hours/week
    [  ] From 5 to 10 hours/week
    [  ] From 10 to 20 hours/week
    [  ] More than 20 hours/week
    - what is/was (are/were) your main motivation(s) to contribute to Open
Source?

    Motivation
    - Why have you tried to contribute to the project? What has motivated you?
      - Were you interest in keep contributing?
            - Was it difficult to find information about this?

    Process
    - What was the first action that you took when decided to join the project?
Which mean have you used?
      - How/why did you choose starting this way?
            - Were you aware of the steps to follow to become a contributor?
            - Did you know how to behave and what were the initial steps?
       Have you used the same approach in other project?
```

**Problems**

- What are your main "frustrations" with the project (considering joining process)?
  - What kind of support were you expecting?
    (- Did you know how to start?
    - Did you know how to find an issue?
    - Did you know how to coordinate your tasks with the team?
  - Were you aware of the organizational structure of the project?
    - Did you know who to contact people to support you?)
  - What kind of mechanism would the project provide to support you?

And, if you could suggest mechanisms or processes to be adopted by the project, what could be your suggestion(s)?

**Process (extra)**

- Have you concluded/delivered any task?
  - How have you selected the task?
  - What were the difficulties faced?
    - Where did you look for support?
    - Have you handled?
    - What kind of support could be helpful in these cases (this case)?
- Can you think about mechanisms that could support newcomers joining process?

# Appendix C
# Open questionnaire sent to projects

In this appendix, we present the email used to recruit the participants and the survey applied to identify barriers faced by newcomers to OSS.

## A.8. Recruitment email

The email sent to the mailing lists and potential answerers was based on the following:

```
Hello,

I am a PhD Candidate from Brazil. My research is interest is on how to support
new contributors during their first steps in the project.

My final goal is to verify what kind of tooling is appropriate to support
the newcomers overcoming their difficulties when they are willing to contribute
to the project.

The first step of my research is to find out what are the main obstacles and
difficulties faced by these newcomers. My goal is to hear from the community
itself.

I need your help answering three quick questions available here:
http://igor.pro.br/limesurvey/index.php?sid=89755&lang=en

You will not take more than 10 minutes, and the survey responses does not
contain any identifying information about you unless a specific question in
the survey has asked for this.

Thanks in advance
```

## A.9. Questionnaire

# Open Source Software Newcomers

This survey aims at finding how to support new contributors of Open Source Software Projects while they are attempting to make their first contribution to the project. The final goal of this research is to verify what kind of support is appropriate for the newcomers to overcome the barriers they face.

The first step is to find out what are the main barriers faced by these newcomers. My goal is to hear from the community itself

The survey should take about 10 minutes, and it is anonymous. All research data collected will be stored securely and confidentially.

Participation in this study is voluntary. There is no cost to you for participating. You may choose to skip a question or a study procedure.

There are 3 questions in this survey

## Survey

**[]To which project do you mainly contribute to?**

Please write your answer here:

(List at most 3)

**[]When did you become a contributor in this project?**

Please choose **only one** of the following:

- ○ Less than 6 months ago
- ○ Between 6 months and 1 year ago
- ○ Between 1 year and 3 years ago
- ○ More than 3 years ago

**[]In your opinion, what are the main barriers faced by newcomer when they want to make their first contribution to this project ?**
**(Consider technical and non-technical issues)**

Please write your answer here:

Please write your answer here:

Thanks for supporting our research!
Your answer will be really important for us!

# Appendix D
# Profiling, self-efficacy and Technology Acceptance Model questionnaires applied to students

In this appendix, we present the instruments used to profile the student newcomers, and to conduct the self-efficacy and TAM assessments. In this sense, we present the questionnaires applied before and after the assignment. The post-assignment questionnaire also brings the questions used to collect the feedback from the students regarding the contribution barriers and regarding the portal (presented only for the group of students who received the credentials to access FLOSScoach)

# A.10. Pre-assignment instrument

Olá,

você está prestes a iniciar sua participação em um estudo sobre a entrada em projetos de software livre. As questões a seguir vão ser usadas para mapear seu perfil. Você não será avaliado por essas respostas ou qualquer informação que seja colocada aqui.

There are 15 questions in this survey

## Perfil

**[]Qual seu nome? ***

Please write your answer here:

**[]Qual sua idade? ***

Please write your answer here:

**[]Qual seu sexo? ***

Please choose **only one** of the following:

- ○ Masculino
- ○ Feminino
- ○ Other

**[]Quanto tempo você tem de experiência em desenvolvimento de software (incluindo experiência acadêmica) ***

Please choose **only one** of the following:

- ○ até 1 ano
- ○ 1 a 2 anos
- ○ 2 a 3 anos
- ○ 3 a 4 anos
- ○ 4 a 5 anos
- ○ 5 a 6 anos
- ○ 6 a 7 anos
- ○ 7 a 8 anos
- ○ 8 a 9 anos
- ○ 9 a 10 anos
- ○ mais de 10 anos

Considere todo o tempo que teve contato com desenvolvimento de software, incluindo curso tecnico, universidade, etc.

**[]Quanto tempo você tem de experiência em desenvolvimento fora da academia? ***

Please choose **only one** of the following:

- ○ até 1 ano
- ○ de 1 a 2 anos
- ○ de 2 a 3 anos
- ○ de 3 a 4 anos
- ○ de 4 a 5 anos
- ○ mais de 5 anos

Considere aqui apenas experiência na indústria (fora da academia). Considere contribuições para projetos de software livre.

**[]Tem experiência em desenvolvimento de projetos de software livre? ***

Please choose **only one** of the following:

- ○ Yes
- ○ No

### []Qual(is) projeto(s)?

**Only answer this question if the following conditions are met:**
Answer was 'Yes' at question '6 [P3]' (Tem experiência em desenvolvimento de projetos de software livre?)

Please write your answer here:

### []Você já pensou em contribuir para um projeto de Software Livre? *

Please choose **only one** of the following:

- ○ Yes
- ○ No

### []Qual foi a motivação?

**Only answer this question if the following conditions are met:**
Answer was 'Yes' at question '8 [P6]' (Você já pensou em contribuir para um projeto de Software Livre?)

Please write your answer here:

### []Contribuiu efetivamente?

**Only answer this question if the following conditions are met:**
Answer was 'Yes' at question '8 [P6]' (Você já pensou em contribuir para um projeto de Software Livre?)

Please choose **only one** of the following:

- ○ Sim
- ○ Não
- ○ Tentei, mas não consegui

### []Experiência em linguagens de programação. *

Please choose the appropriate response for each item:

| | Não tenho experiência | Conheço pouco | Conheço razoavelmente | Conheço bem | Sou experiente |
|---|---|---|---|---|---|
| C | ○ | ○ | ○ | ○ | ○ |
| C++ | ○ | ○ | ○ | ○ | ○ |
| Java | ○ | ○ | ○ | ○ | ○ |
| Python | ○ | ○ | ○ | ○ | ○ |
| SQL | ○ | ○ | ○ | ○ | ○ |
| Perl | ○ | ○ | ○ | ○ | ○ |
| Shell Script | ○ | ○ | ○ | ○ | ○ |

### []Experiência em tecnologias usadas nos projetos *

Please choose the appropriate response for each item:

| | Não tenho experiência | Conheço pouco | Conheço razoavelmente | Conheço bem | Sou experiente |
|---|---|---|---|---|---|
| Git | ○ | ○ | ○ | ○ | ○ |
| SVN | ○ | ○ | ○ | ○ | ○ |
| Gerrit | ○ | ○ | ○ | ○ | ○ |
| Linha de Comando | ○ | ○ | ○ | ○ | ○ |
| Linux | ○ | ○ | ○ | ○ | ○ |
| Bugzilla | ○ | ○ | ○ | ○ | ○ |

**[]Contribuiu efetivamente?**

**Only answer this question if the following conditions are met:**
Answer was 'Yes' at question '8 [P6]' (Você já pensou em contribuir para um projeto de Software Livre?)

Please choose **only one** of the following:

- ○ Sim
- ○ Não
- ○ Tentei, mas não consegui

**[]Experiência em linguagens de programação. ***

Please choose the appropriate response for each item:

| | Não tenho experiência | Conheço pouco | Conheço razoavelmente | Conheço bem | Sou experiente |
|---|---|---|---|---|---|
| C | ○ | ○ | ○ | ○ | ○ |
| C++ | ○ | ○ | ○ | ○ | ○ |
| Java | ○ | ○ | ○ | ○ | ○ |
| Python | ○ | ○ | ○ | ○ | ○ |
| SQL | ○ | ○ | ○ | ○ | ○ |
| Perl | ○ | ○ | ○ | ○ | ○ |
| Shell Script | ○ | ○ | ○ | ○ | ○ |

**[]Experiência em tecnologias usadas nos projetos ***

Please choose the appropriate response for each item:

| | Não tenho experiência | Conheço pouco | Conheço razoavelmente | Conheço bem | Sou experiente |
|---|---|---|---|---|---|
| Git | ○ | ○ | ○ | ○ | ○ |
| SVN | ○ | ○ | ○ | ○ | ○ |
| Gerrit | ○ | ○ | ○ | ○ | ○ |
| Linha de Comando | ○ | ○ | ○ | ○ | ○ |
| Linux | ○ | ○ | ○ | ○ | ○ |
| Bugzilla | ○ | ○ | ○ | ○ | ○ |

## Self Efficacy (round 1)

**[]Por favor indique o quão verdadeiro é cada um dos seguintes comentários para você ***

Please choose the appropriate response for each item:

| | Discordo (1) | (2) | De certa forma (3) | (4) | Totalmente (5) |
|---|---|---|---|---|---|
| Sinto-me confortável pedindo ajuda a uma comunidade utilizando meios de comunicação eletrônicos | ○ | ○ | ○ | ○ | ○ |
| Posso escrever minhas dúvidas e entender as respostas em inglês | ○ | ○ | ○ | ○ | ○ |
| Consigo entender código fonte escrito por outras pessoas | ○ | ○ | ○ | ○ | ○ |
| Tenho boas habilidades para escrever e alterar código fonte | ○ | ○ | ○ | ○ | ○ |
| Sinto-me confortável com o processo de contribuição de um projeto de software livre | ○ | ○ | ○ | ○ | ○ |
| Acredito que contribuir para um projeto de software livre seja uma atividade interessante | ○ | ○ | ○ | ○ | ○ |
| Sinto que posso obter o código fonte, configurar e rodar uma aplicação se as instruções apropriadas forem fornecidas | ○ | ○ | ○ | ○ | ○ |
| Acredito que consigo entender bugs relatados pela comunidade e encontrar soluções para eles | ○ | ○ | ○ | ○ | ○ |
| Eu consigo escolher uma tarefa que eu seja capaz de executar em um projeto de software livre, caso uma lista de tarefas seja fornecida | ○ | ○ | ○ | ○ | ○ |
| Sou capaz de encontrar no código fonte quais o(s) local(is) que devo alterar para corrigir um bug relatado por terceiros | ○ | ○ | ○ | ○ | ○ |

Obrigado!

Agora, vamos iniciar nosso trabalho.

Boa sorte

# A.11. Post-assignment instrument

## [FC] Feedback e Self-efficacy pós-contribuição

Olá,

você está prestes a iniciar sua participação em um estudo sobre a entrada em projetos de software livre. As questões a seguir vão ser usadas para mapear seu perfil. Você não será avaliado por essas respostas ou qualquer informação que seja colocada aqui.

There are 12 questions in this survey

### Self Efficacy (round 2)

**[]Qual seu nome? ***

Please write your answer here:

Digite seu nome completo

**[]Por favor indique o quão verdadeiro é cada um dos seguintes comentários para você ***

Please choose the appropriate response for each item:

| | Não se aplica (1) | (2) | De certa forma (3) | (4) | Totalmente (5) |
|---|---|---|---|---|---|
| Sinto-me confortável pedindo ajuda a uma comunidade (desconhecidos) utilizando meios de comunicação eletrônicos | ○ | ○ | ○ | ○ | ○ |
| Posso escrever minhas dúvidas e entender as respostas em inglês | ○ | ○ | ○ | ○ | ○ |
| Consigo entender código fonte escrito por outras pessoas | ○ | ○ | ○ | ○ | ○ |
| Tenho boas habilidades para escrever e alterar código fonte | ○ | ○ | ○ | ○ | ○ |
| Sinto-me confortável em tentar contribuir para um projeto de software livre | ○ | ○ | ○ | ○ | ○ |
| Acredito que contribuir para um projeto de software livre seja uma atividade interessante | ○ | ○ | ○ | ○ | ○ |
| Acredito que consigo entender bugs relatados pela comunidade e encontrar soluções para eles | ○ | ○ | ○ | ○ | ○ |
| Sinto que posso obter o código fonte, configurar e rodar uma aplicação se as instruções apropriadas forem fornecidas | ○ | ○ | ○ | ○ | ○ |
| Eu consigo escolher uma tarefa que eu seja capaz de executar em um projeto de software livre, caso uma lista de tarefas seja fornecida | ○ | ○ | ○ | ○ | ○ |
| Sou capaz de encontrar no código fonte quais o(s) local(is) que devo alterar para corrigir um bug relatado por terceiros | ○ | ○ | ○ | ○ | ○ |

### Feedback

**[]**

**Como você descreveria sua experiência com essa atividade? Identifique quais foram os pontos positivos e negativos do processo de contribuição ***

Please write your answer here:

**[]Descreva seu sentimento com relação ao seu resultado ao final dessa atividade. ***

Please write your answer here:

**[]Quais foram as principais barreiras que você encontrou durante a condução da atividade? Detalhe. ***

Please write your answer here:

**[]Teve alguma interação social com a comunidade (email, IRC etc.)? Como foi a experiência? ***

Please write your answer here:

Please write your answer here:

**[]O que você sugeriria para que a comunidade a fim de melhorar a experiência de novatos que desejem contribuir? (ferramentas, estratégias, informações etc.) ***

Please write your answer here:

**[]Como você descreve o papel do flosscoach no processo de contribuição? ***

Please write your answer here:

[]

**O que considerou ruim ou desnecessário no portal flosscoach?**

*

Please write your answer here:

[]**Tem alguma sugestão de melhoria para o portal flosscoach?**

Please write your answer here:

## TAM - Sobre o portal flosscoach

**[]Selecione a opção que melhor represente sua opinião com relação ao portal flosscoach ***

Please choose the appropriate response for each item:

| | Discordo completamente | Discordo | Discordo parcialmente | Concordo parcialmente | Concordo | Concordo completamente |
|---|---|---|---|---|---|---|
| Usando um portal como o flosscoach para contribuir para um novo projeto de software livre, eu seria capaz de contribuir de forma mais rápida. | ○ | ○ | ○ | ○ | ○ | ○ |
| Usando um portal como o flosscoach pode melhorar o desempenho dos novatos em projetos de software livre | ○ | ○ | ○ | ○ | ○ | ○ |
| Usando um portal como o flosscoach permitiria aos novatos aumentar a sua produtividade | ○ | ○ | ○ | ○ | ○ | ○ |
| Usando um portal como o flosscoach aumentaria a eficácia dos novatos em colaborar. | ○ | ○ | ○ | ○ | ○ | ○ |
| Usando um portal como o flosscoach tornaria a tarefa de contribuir para projetos de software livre mais fácil para novatos | ○ | ○ | ○ | ○ | ○ | ○ |
| Eu acho que um portal como o flosscoach é útil para novatos desejando contribuir para um projeto de OSS | ○ | ○ | ○ | ○ | ○ | ○ |
| Aprender a operar o flosscoach foi fácil para mim | ○ | ○ | ○ | ○ | ○ | ○ |
| Eu achei fácil fazer o flosscoach aquilo que eu gostaria que ele fizesse, para apoiar os primeiros passos de novatos que desejemcontribuir para um projeto de software livre | ○ | ○ | ○ | ○ | ○ | ○ |
| Minha interação com o flosscoach foi clara e compreensível. | ○ | ○ | ○ | ○ | ○ | ○ |
| Seria fácil tornar-se hábil em usar o flosscoach | ○ | ○ | ○ | ○ | ○ | ○ |
| É fácil lembrar como realizar tarefas usando flosscoach | ○ | ○ | ○ | ○ | ○ | ○ |
| Eu acho o flosscoach fácil de usar | ○ | ○ | ○ | ○ | ○ | ○ |
| Assumindo que o flosscoach estivesse disponível para qualquer projeto, eu o usaria no futuro | ○ | ○ | ○ | ○ | ○ | ○ |
| Eu preferiria usar flosscoach às páginas do projeto para guiar-me na contribuição para um projeto de software livre | ○ | ○ | ○ | ○ | ○ | ○ |

Obrigado!

Agora, vamos iniciar nosso trabalho.

# Appendix E

## Analysis of the Barriers Regarding Project Characteristics

In this appendix we present the complementary material to the analysis of the barriers per project and according to the projects characteristics. In the following table we bring the summary of the analysis per project. The number presented in each cell corresponds to the number of participants who mentioned each barrier. The number include the analysis of the feedback from students, open questions and interviews with community members. We present these numbers just to illustrate the how widely a barrier was faced or knew by the participants.

| Category/Subcategory | Barrier | aTunes | Audacity | cogroo | Etherpad | FreeMind | Gephi | Integrade | JabRef | jEdit | LibreOffice | Moodle | Mozilla Firefox | Noosfero | Open Office | OpenVPN | Pardus | Text | Zxing |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reception Issues | Not receiving an answer | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Delayed answers | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| | Impolite answers | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 |
| | Receiving answers with too advanced/complex contents | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Newcomers' characteristics / Newcomers' behavior | Lack of Commitment | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | Underestimating the challenge | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Lack of proactivity | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 2 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| | Lack of patience | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Newcomers' characteristics / Newcomers' behavior / Newcomers' communication | Newcomers do not acknowledge/thank answers received | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Shyness | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| | English level | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Making useless comments in the mailing list/forums | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| | Low responsiveness | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Not sending a correct meaningful/correct message | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Newcomers' characteristics / Newcomers previous knowledge | Lack of domain expertise | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Lack of knowledge in project process and practices | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Newcomers' characteristics / Newcomers' previous knowledge / Newcomers' technical background | Knowledge on technologies and tools used by the project | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 3 | 0 | 2 | 1 | 2 | 0 | 0 | 1 | 0 | 0 | 0 |
| | Knowledge on versioning control system | 1 | 0 | 0 | 2 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| | Choosing the right development tools | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Proper knowledge in the programming language | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 2 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Experience on unit testing | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Newcomers' orientation | Finding a task to start with | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 4 | 0 | 2 | 1 | 0 | 0 | 1 | 0 | 1 |
| | Finding the correct artifacts to fix an issue | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 3 | 1 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Outdated list of bugs | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Reproducing issues | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Finding a mentor | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 2 | 1 | 2 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Poor “How to contribute” available | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 1 | 0 | 0 | 1 | 2 | 1 | 0 | 0 | 0 |
| | Newcomers don’t know what is the contribution flow | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Documentation problems | Outdated documentation | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 4 | 0 | 2 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| | Information overload | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Unclear documentation | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Spread documentation | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Code comments not clear | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Documentation problems / Lack of documentation | Documentation in general | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 2 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| | Design documents | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Documentation on project structure | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Documentation on setting up workspace | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 2 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Code comments | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| | Code documentation | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Technical hurdles / Code /architecture hurdles / Code characteristics | Code complexity/instability | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Bad design quality | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Bad code quality | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 2 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Outdated code | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Codebase size | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 2 | 0 | 3 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 |
| | Lack of code standards | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Technical hurdles / Code /architecture hurdles / Cognitive problems | Understanding architecture/code structure | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 2 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| | Understanding the code | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 4 | 0 | 3 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| | Understanding flow of information | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Technical hurdles / Change request hurdles | Delay to get contribution accepted/reviewed | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 2 | 0 | 0 | 1 | 0 | 0 |
| | Getting contribution accepted | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Lack of information on how to send a contribution | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| | Issue to create a patch | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Technical hurdles / Local environment setup hurdles | Building workspace locally | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 4 | 2 | 5 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 |
| | Platform dependency | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Finding the correct source | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | Library dependency | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Cultural differences | Some newcomers need to contact a real person | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| | Message received is considered rude | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |

In the next table, we present the amount of participants who mentioned each barrier, now grouping them by project characteristic. By presenting this data our goal is to enable comparisons cross-characteristics.

| | | Size (in Lines of Code) | | | | Main programming language | | | | | | | Age of the project | | | Hosting forge | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | > 1,000 KLoC | 200 - 500 KLoC | 100 - 200 KLoC | < 100 KLoC | Java | C++ | PHP | JS | Ruby | C | Multiple | > 10 years | 6 to 10 years | <= 5 years | Github/GitLab | SourceForge | Own Hosting | LaunchPad / GoogleCode |
| | # of projects --> | 5 | 4 | 6 | 3 | 9 | 4 | 1 | 1 | 1 | 1 | 1 | 8 | 9 | 2 | 6 | 5 | 6 | 2 |
| **Category/Subcategory** | **Barrier** | | | | | | | | | | | | | | | | | | |
| Reception Issues | Not receiving an answer | | | 2 | | 2 | | | | | | | 2 | | | | 2 | | |
| | Delayed answers | | 1 | 2 | 1 | 3 | | | | 1 | | | 2 | 2 | | 2 | 2 | | |
| | Impolite answers | 2 | 1 | | 1 | | | | 1 | 1 | | 1 | 1 | 2 | 1 | 2 | | 2 | |
| | Receiving answers with too advanced/complex contents | 2 | 1 | | 1 | 1 | 3 | | | | | | 3 | 1 | | 1 | | 2 | 1 |
| Newcomers' characteristics / Newcomers' behavior | Lack of Commitment | 1 | 1 | 2 | | 2 | 1 | | | 1 | | | 2 | 1 | 1 | 1 | 2 | 1 | |
| | Underestimating the challenge | 1 | | | | | 1 | | | | | | 1 | | | | | 1 | |
| | Lack of proactivity | 4 | | 1 | | 1 | 3 | 1 | | | | | 5 | | | | | 4 | 1 |
| | Lack of patience | 2 | | | | | 1 | 1 | | | | | 2 | | | | | 2 | |
| Newcomers' characteristics / Newcomers' behavior / Newcomers' communication | Newcomers do not acknowledge/thank answers received | | | 1 | | 1 | | | | | | | 1 | | | | 1 | | |
| | Shyness | 1 | 1 | | | | | | | 1 | | | | 2 | | 1 | | 1 | |
| | English level | 2 | 1 | 1 | | 2 | 2 | | | | | | 4 | | | | 2 | 2 | |
| | Making useless comments in the mailing list/forums | 1 | | | | | | | | | | | | 1 | | | | 1 | |
| | Low responsiveness | | | 1 | | 1 | | | | | | | 1 | | | | 1 | | |
| | Not sending a correct meaningful/correct message | | 1 | 1 | | 2 | | | | | | | 2 | | | | 2 | | |
| Newcomers' characteristics / Newcomers previous knowledge | Lack of domain expertise | | | | | | | | | | | | | | | | | | |
| | Lack of knowledge in project process and practices | | | | | | | | | | | | | | | | | | |
| Newcomers' characteristics / Newcomers' previous knowledge / Newcomers' technical background | Knowledge on technologies and tools used by the project | 5 | 1 | 5 | 1 | 5 | 2 | 1 | 1 | | 1 | 2 | 8 | 3 | 1 | 3 | 4 | 5 | |
| | Knowledge on versioning control system | 1 | 2 | 3 | 2 | 4 | | 1 | 2 | | 1 | | 3 | 3 | 2 | 3 | 4 | 1 | |
| | Choosing the right development tools | | 1 | | 1 | | 1 | | 1 | | | | 1 | | 1 | 1 | | | 1 |
| | Proper knowledge in the programming language | 3 | | 4 | 1 | 4 | 3 | | 1 | | | | 5 | 2 | 1 | 3 | 2 | 3 | |
| | Experience on unit testing | | | 1 | | 1 | | | | | | | | 1 | | 1 | | | |
| Newcomers' orientation | Finding a task to start with | 7 | 1 | 3 | 1 | 5 | 4 | | | 1 | | 2 | 8 | 5 | | 2 | 3 | 8 | |
| | Finding the correct artifacts to fix an issue | 2 | 1 | 5 | | 6 | 2 | | | | | | 6 | 2 | | 1 | 5 | 2 | |
| | Outdated list of bugs | 2 | | 1 | | 1 | 1 | | | | | 1 | 3 | | | | 1 | 2 | |

| | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Reproducing issues | | | | | 1 | | | | | | | | 1 | | | | | |
| | Finding a mentor | 5 | | 3 | 1 | 4 | 2 | 1 | 1 | | | 2 | 7 | 2 | 1 | 1 | 3 | 6 | |
| | Poor “How to contribute” available | 3 | 2 | 2 | 1 | 2 | 3 | | 1 | 1 | 1 | | 5 | 2 | 1 | 2 | 3 | 3 | |
| | Newcomers don’t know what is the contribution flow | 1 | | | | | 1 | | | | | | 1 | | | | | 1 | |
| Documentation problems | Outdated documentation | 4 | | 4 | 1 | 4 | 3 | | 1 | | | 1 | 8 | | 1 | 1 | 4 | 4 | |
| | Information overload | | | | | | | | | | | | | | | | | | |
| | Unclear documentation | 2 | | | | | 2 | | | | | | 2 | | | | | 2 | |
| | Spread documentation | 3 | | | | | 2 | 1 | | | | | 3 | | | | | 3 | |
| | Code comments not clear | 1 | | | | | 1 | | | | | | 1 | | | | | 1 | |
| Documentation problems / Lack of documentation | Documentation in general | 4 | | 1 | 1 | 1 | 2 | | 1 | | | 1 | 4 | 1 | 1 | 1 | 1 | 4 | |
| | Design documents | 2 | | 3 | | 3 | 1 | | | | | 1 | 4 | 1 | | | 3 | 2 | |
| | Documentation on project structure | 1 | | | | | | | | | | 1 | 1 | | | | | 1 | |
| | Documentation on setting up workspace | 2 | | 3 | | 3 | 2 | | | | | | 5 | | | | 2 | 2 | 1 |
| | Code comments | | 1 | 2 | 1 | 2 | 2 | | | | | | 3 | 1 | | 1 | 2 | | 1 |
| | Code documentation | | 1 | 3 | | 3 | 1 | | | | | | 4 | | | | 3 | | 1 |
| Technical hurdles / Code /architecture hurdles / Code characteristics | Code complexity/instability | 2 | | | | | 2 | | | | | | 2 | | | | | 2 | |
| | Bad design quality | 1 | | 2 | 1 | 2 | 1 | | 1 | | | | 3 | | 1 | 1 | 2 | 1 | |
| | Bad code quality | 3 | | 3 | 1 | 3 | 3 | | 1 | | | | 5 | | 2 | 1 | 3 | 3 | |
| | Outdated code | | 1 | | | | 1 | | | | | | 1 | | | | | | 1 |
| | Codebase size | 5 | | 2 | 1 | 2 | 4 | | 1 | | | 1 | 7 | | 1 | 1 | 2 | 5 | |
| | Lack of code standards | 1 | | 1 | | 1 | 1 | | | | | | 2 | | | | 1 | 1 | |
| Technical hurdles / Code /architecture hurdles / Cognitive problems | Understanding architecture/code structure | 1 | 1 | 3 | 1 | 5 | 1 | | | | | | 4 | 2 | | 2 | 3 | 1 | |
| | Understanding the code | 5 | 2 | 4 | 2 | 4 | 6 | 1 | 1 | 1 | | | 10 | 2 | 1 | 3 | 4 | 5 | 1 |
| | Understanding flow of information | | | 1 | | 1 | | | | | | | 1 | | | | 1 | | |
| Technical hurdles / Change request hurdles | Delay to get contribution accepted/reviewed | 2 | 3 | 1 | | 1 | 2 | | | 2 | | | 2 | 3 | | 2 | 1 | 2 | |
| | Getting contribution accepted | | 1 | 1 | | 2 | 2 | | | | | | 1 | 1 | | | 1 | 1 | |
| | Lack of information on how to send a contribution | 1 | 3 | 1 | 2 | 2 | 3 | | 2 | 1 | | | 3 | 1 | 2 | 3 | 2 | 1 | |
| | Issue to create a patch | 1 | 1 | | | 1 | 2 | | | | | | 2 | | | | 1 | 1 | |
| Technical hurdles / Local environment setup hurdles | Building workspace locally | 6 | 13 | 8 | 2 | 11 | 19 | | 1 | 1 | | | 14 | 5 | 1 | 5 | 7 | 6 | 2 |
| | Platform dependency | 1 | 3 | 1 | | 2 | 4 | | | | | | 2 | 1 | | | 2 | 1 | |
| | Finding the correct source | 1 | 1 | 1 | | 1 | 2 | | | | | | 2 | | | | 1 | 1 | |
| | Library dependency | 2 | 1 | 1 | | 1 | 3 | | | | | | 3 | | | | 1 | 2 | |
| Cultural differences | Some newcomers need to contact a real person | 1 | 1 | | | | | | | 1 | | | | 2 | | 1 | | 1 | |
| | Message received is considered rude | 2 | | | | | 1 | | | | | | 1 | 1 | | | | 2 | |

# References


Adler, A., Gujar, A., Harrison, B.L., O'Hara, K., Sellen, A. 1998. A Diary Study of Work-related Reading: Design Implications for Digital Reading Devices. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Los Angeles, California, USA, 1998), 241–248.

Arguello, J., Butler, B.S., Joyce, E., Kraut, R., Ling, K.S., RosÃ¨, C., Wang, X. 2006. Talk to Me: Foundations for Successful Individual-group Interactions in Online Communities. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Montréal, Quebec, Canada, 2006), 959–968.

Babar, M.A., Winkler, D., Biffl, S. 2007. Evaluating the Usefulness and Ease of Use of a Groupware Tool for the Software Architecture Evaluation Process. *Proceedings of the First International Symposium on Empirical Software Engineering and Measurement* (Sep. 2007), 430–439.

Baier, S.T. 2005. *International Students: Culture Shock and Adaptation to the US Culture*. Eastern Michigan University.

Bandura, A. 1977. Self-efficacy: toward a unifying theory of behavioral change. *Psychological review*. 84, 2 (1977), 191.

Bandura, A. 1986. *Social foundations of thought and action: a social cognitive theory*. Prentice-Hall.

Beecham, S., Baddoo, N., Hall, T., Robinson, H., Sharp, H. 2008. Motivation in Software Engineering: A systematic literature review. *Information and Software Technology*. 50, 9–10 (Aug. 2008), 860–878.

Begel, A., Simon, B. 2008. Novice Software Developers, All over Again. *Proceedings of the Fourth international Workshop on Computing Education Research* (2008), 3–14.

Ben, X., Beijun, S., Weicheng, Y. 2013. Mining Developer Contribution in Open Source Software Using Visualization Techniques. *Proceedings of the 2013 Third International Conference on Intelligent System Design and Engineering Applications* (Jan. 2013), 934–937.

Berlin, L.M. 1992. *Beyond Program Understanding: A Look at Programming Expertise in Industry*. Hewlett-Packard Laboratories.

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.

Bird, C., Gourley, A., Devanbu, P., Swaminathan, A., Hsu, G. 2007. Open Borders? Immigration in Open Source Projects. *Proceedings of the Fourth International Workshop on Mining Software Repositories* (2007), 6–6.

Bonaccorsi, A., Rossi, C. 2004. Altruistic individuals, selfish firms? The structure of motivation in Open Source software. *First Monday*. 9, 1 (Jan. 2004), [online].

Brooks, F.P. 1995. *The Mythical Man-Month: Essays on Software Engineering*. Addison-Wesley Professional.

Bryant, S.L., Forte, A., Bruckman, A. 2005. Becoming Wikipedian: Transformation of Participation in a Collaborative Online Encyclopedia. *Proceedings of the 2005 International ACM SIGGROUP Conference on Supporting Group Work* (Sanibel Island, Florida, USA, 2005), 1–10.

Burke, M., Joyce, E., Kim, T., Anand, V., Kraut, R. 2007. Introductions and Requests: Rhetorical Strategies That Elicit Response in Online Communities. *Communities and Technologies 2007*. Springer- Verlag London. 21–39.

Burke, M., Marlow, C., Lento, T. 2009. Feed Me: Motivating Newcomer Contribution in Social Network Sites. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Boston, MA, USA, 2009), 945–954.

Canfora, G., Penta, M. di, Oliveto, R., 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.

Capiluppi, A., Adams, P.J. 2009. Reassessing Brooks Law for the Free Software Community. *Open Source Ecosystems: Diverse Communities Interacting*. Springer Berlin Heidelberg. 274–283.

Capiluppi, A., Michlmayr, M.F. 2007. From the Cathedral to the Bazaar: An Empirical Study of the Lifecycle of Volunteer Community Projects. *Open Source Development, Adoption and Innovation*. Springer-Verlag Boston. 31–44.

Capra, E., Wasserman, A.I. 2008. A Framework for Evaluating Managerial Styles in Open Source Projects. *Open Source Development, Communities and Quality*. Springer. 1–14.

Carmines, E.G., Zeller, R.A. 1979. *Reliability and Validity Assessment*. SAGE Publications.

Cassidy, S., Eachus, P. 2002. Developing the computer user self-efficacy (CUSE) scale: investigating the relationship between computer self-efficacy, gender and experience with computers. *Journal of Educational Computing Research*. 26, 2 (Jan. 2002), 133–153.

Chengalur-Smith, I.N., Sidorova, A., Daniel, S.L. 2010. Sustainability of Free/Libre Open Source Projects: A Longitudinal Study. *Journal of the Association for Information Systems*. 11, 11 (Nov. 2010), 657–683.

Choi, B., Alexander, K., Kraut, R.E., Levine, J.M. 2010. Socialization Tactics in Wikipedia and Their Effects. *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work* (Savannah, Georgia, USA, 2010), 107–116.

Choi, B.R. 2012. *Essays on Socialization in Online Groups*. Tepper School of Business - Carneggie Mellon University.

Cockburn, A. 2000. Selecting a Project's Methodology. *IEEE Software*. 17, 4 (Jul. 2000), 64–71.

Colazo, J., Fang, Y. 2009. Impact of License Choice on Open Source Software Development Activity. *Journal of the American Society for Information Science and Technology*. 60, 5 (May. 2009), 997–1011.

Costa Valentim, N.M., Conte, T. 2014. Improving a Usability Inspection Technique Based on Quantitative and Qualitative Analysis. *2014 Brazilian Symposium on Software Engineering (SBES)* (2014), 171–180.

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

Cubranic, D., Murphy, G.C., Singer, J., Booth, K.S. 2005. Hipikat: a project memory for software development. *IEEE Transactions on Software Engineering*. 31, 6 (Jun. 2005), 446–465.

Czerwinski, M., Horvitz, E., Wilhite, S. 2004. A Diary Study of Task Switching and Interruptions. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vienna, Austria, 2004), 175–182.

Dagenais, B., Ossher, H., Bellamy, R.K.E., Robillard, M.P., Vries, J.P. de 2010. Moving into a new software project landscape. *Proceedings of the 2010 ACM/IEEE 32nd International Conference on Software Engineering* (New York, NY, USA, 2010), 275–284.

David, P.A., Shapiro, J.S. 2008. Community-based production of open-source software: What do we know about the developers who participate? *Information Economics and Policy*. 20, 4 (Dec. 2008), 364–398.

Davidson, J.L., Mannan, U.A., Naik, R., Dua, I., Jensen, C. 2014. Older Adults and Free/Open Source Software: A Diary Study of First-Time Contributors. *Proceedings of The International Symposium on Open Collaboration* (2014), A5.

Davis, F.D. 1989. Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. *MIS Quarterly*. 13, 3 (Sep. 1989), 319–340.

Dittrich, Y. 2014. Software engineering beyond the project – Sustaining software ecosystems. *Information and Software Technology*. 56, 11 (2014), 1436–1456.

Dittrich, Y., John, M., Singer, J., Tessem, B. 2007. Editorial: For the Special Issue on Qualitative Software Engineering Research. *Information and Software Technology*. 49, 6 (Jun. 2007), 531–539.

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

Fagerholm, F., Johnson, P., Guinea, A.S., Borenstein, J., Münch, J. 2014. Onboarding in Open Source Projects. *IEEE Software*. 31, 6 (Nov. 2014), 54–61.

Fang, Y., Neufeld, D. 2009. Understanding Sustained Participation in Open Source Software Projects. *Journal of Management Information Systems*. 25, 4 (Apr. 2009), 9–50.

Farzan, R., Kraut, R.E. 2013. Wikipedia Classroom Experiment: Bidirectional Benefits of Students' Engagement in Online Production Communities. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France, 2013), 783–792.

Faulkner, R., Walling, S., Pinchuk, M. 2012. Etiquette in Wikipedia: Weening New Editors into Productive Ones. *Proceedings of the Eighth Annual International Symposium on Wikis and Open Collaboration* (Linz, Austria, 2012), Article 5.

Fishbein, M., Ajzen, I. 1975. *Belief, attitude, intention and behavior: An introduction to theory and research*. Addison-Wesley Pub.

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

Foner, N., Alba, R. 2008. Immigrant Religion in the U.S. and Western Europe: Bridge or Barrier to Inclusion? *International Migration Review*. 42, 2 (May. 2008), 360–392.

Forte, A., Lampe, C. 2013. Defining, Understanding, and Supporting Open Collaboration: Lessons From the Literature. *American Behavioral Scientist*. 57, 5 (Jan. 2013), 535–547.

França, A.C.C., Gouveia, T.B., Santos, P.C.F., Santana, C.A., Silva, F.Q.B. da 2011. Motivation in software engineering: A systematic review update. *Proceedings of the 15th Annual Conference on Evaluation Assessment in Software Engineering* (Apr. 2011), 154–163.

França, A.C.C., Sharp, H., Silva, F.Q.B. da 2014a. Motivated Software Engineers Are Engaged and Focused, While Satisfied Ones Are Happy. *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement* (Torino, Italy, 2014), 32:1–32:8.

França, A.C.C., Silva, F.Q.B. da 2010. Designing Motivation Strategies for Software Engineering Teams: An Empirical Study. *Proceedings of the 2010 ICSE Workshop on Cooperative and Human Aspects of Software Engineering* (Cape Town, South Africa, 2010), 84–91.

França, A.C.C., Silva, F.Q.B. da, L. C. Felix, A. de, Carneiro, D.E.S. 2014b. Motivation in software engineering industrial practice: A cross-case analysis of two software organisations. *Information and Software Technology*. 56, 1 (Jan. 2014), 79–101.

França, A.C.C., Silva, da Fabio Q. B. da 2009. An Empirical Study on Software Engineers Motivational Factors. *Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement* (Washington, DC, USA, 2009), 405–409.

Fugelstad, P., Dwyer, P., Moses, J.F., Kim, J., Mannino, C.A., Terveen, L., Snyder, M. 2012. What Makes Users Rate (Share, Tag, Edit...)?: Predicting Patterns of Participation in Online Communities. *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work* (Seattle, Washington, USA, 2012), 969–978.

Fuks, H., Raposo, A.B., Gerosa, M.A., Lucena, C.J.P. 2005. Applying the 3C model to Groupware Development. *International Journal of Cooperative Information Systems*. 14, 2–3 (2005), 299–328.

Fuks, H., Raposo, A.B., Gerosa, M.A., Pimentel, M., Lucena, C.J.P. de 2007. The 3C Collaboration Model. *The encyclopedia of e-Collaboration*. Information Science Reference. 637–644.

Gutwin, C., Greenberg, S., Roseman, M. 1996. Workspace Awareness in Real-Time Distributed Groupware: Framework, Widgets, and Evaluation. *Proceedings of HCI on People and Computers XI* (London, UK, 1996), 281–298.

Halfaker, A., Geiger, R.S., Morgan, J., Riedl, J. 2013a. The Rise and Decline of an Open Collaboration System: How Wikipedia's reaction to sudden popularity is causing its decline. *American Behavioral Scientist*. 57, 5 (May. 2013), 664–688.

Halfaker, A., Keyes, O., Taraborelli, D. 2013b. Making Peripheral Participation Legitimate: Reader Engagement Experiments in Wikipedia. *Proceedings of the 2013 Conference on Computer Supported Cooperative Work* (San Antonio, Texas, USA, 2013), 849–860.

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

Hannebauer, C., Book, M., Gruhn, V. 2014. An Exploratory Study of Contribution Barriers Experienced by Newcomers to Open Source Software Projects. *Proceedings of the First International Workshop on CrowdSourcing in Software Engineering* (Hyderabad, India, 2014), 11–14.

Hars, A., Ou, S. 2001. Working for free? Motivations of participating in open source projects. *Proceedings of the 34th Annual Hawaii International Conference on System Sciences* (2001), 1–9.

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

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

Hertel, G., Niedner, S., Herrmann, S. 2003. Motivation of software developers in Open Source projects: an Internet-based survey of contributors to the Linux kernel. *Research Policy*. 32, 7 (Jul. 2003), 1159–1177.

Hess, J., Wulf, V. 2009. Explore Social Behavior Around Rich-media: A Structured Diary Study. *Proceedings of the Seventh European Conference on European Interactive Television Conference* (Leuven, Belgium, 2009), 215–218.

Hinchcliffe, V., Gavin, H. 2009. Social and Virtual Networks: Evaluating Synchronous Online Interviewing Using Instant Messenger. *The Qualitative Report*. 14, 2 (Jun. 2009), 318–340.

Hoda, R., Noble, J., Marshall, S. 2010. Using Grounded Theory to Study the Human Aspects of Software Engineering. *Proceedings of the Human Aspects of Software Engineering* (Reno, Nevada, 2010), Article 5.

Höst, M., Regnell, B., Wohlin, C. 2000. Using Students as Subjects – A Comparative Study of Students and Professionals in Lead-Time Impact Assessment. *Empirical Software Engineering*. 5, 3 (Nov. 2000), 201–214.

Hsieh, G., Hou, Y., Chen, I., Truong, K.N. 2013. Welcome!": Social and Psychological Predictors of Volunteer Socializers in Online Communities. *Proceedings of the 2013 Conference on Computer Supported Cooperative Work* (San Antonio, Texas, USA, 2013), 827–838.

Jalali, S., Wohlin, C. 2012. Systematic Literature Studies: Database Searches vs. Backward Snowballing. *Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement* (Lund, Sweden, 2012), 29–38.

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

Jepsen, L.O., Mathiassen, L., Nielsen, P.A. 1998. Using Diaries. *Reflective Systems Development*. Aalborg University.

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

Jergensen, N. 2007. Developer autonomy in the FreeBSD open source project. *Journal of Management and Governance*. 11, 2 (May. 2007), 119–128.

Jin, X.-L., Lee, M.K.O., Cheung, C.M.K. 2010. Predicting continuance in online communities: model development and empirical test. *Behaviour and Information Technology*. 29, 4 (Jul. 2010), 383–394.

Johnson, J.P. 2001. Economics of Open Source Software. *Unpublished paper*.

Joyce, E., Kraut, R.E. 2006. Predicting Continued Participation In Newsgroups. *Journal of Computer-Mediated Communication*. 11, 3 (Apr. 2006), 723–747.

Kanfer, R. 1990. Motivation Theory and Industrial and Organizational Psychology. *Handbook of Psychology, Industrial and Organizational Psychology*. Counsulting Psychologist Press.

Ke, W., Zhang, P. 2010. The Effects of Extrinsic Motivations and Satisfaction in Open Source Software Development. *Journal of the Association for Information Systems*. 11, 12 (Dec. 2010), 784–808.

Kemmis, S., McTaggart, R., Nixon, R. 2014. *The action research planner*. Springer Singapore.

Kersten, M., Murphy, G.C. 2005. Mylar: A Degree-of-interest Model for IDEs. *Proceedings of the 4th International Conference on Aspect-oriented Software Development* (Chicago, Illinois, 2005), 159–168.

King, W.R., He, J. 2006. A meta-analysis of the technology acceptance model. *Information and Management*. 43, 6 (Sep. 2006), 740–755.

Kitchenham, B. 2004. *Procedures for Performing Systematic Reviews*. Technical Report #TR/SE-0401. Departament of Computer Science, Keele University.

Kitchenham, B., Brereton, P. 2013. A systematic review of systematic review process research in software engineering. *Information and Software Technology*. 55, 12 (Dec. 2013), 2049–2075.

Kitchenham, B., Charters, S. 2007. *Guidelines for performing Systematic Literature Reviews in Software Engineering*. Technical Report #EBSE 2007-001. Keele University and Durham University.

Koch, S. 2004. Profiling an Open Source Project Ecology and Its Programmers. *Electronic Markets*. 14, 2 (May. 2004), 77–88.

Kraut, R.E., Burke, M., Riedl, J., Resnick, P. 2012. The Challenges of Dealing with Newcomers. *Building Successful Online Communities: Evidence-Based Social Design*. MIT Press. 179–230.

Krogh, G. von, Haefliger, S., Spaeth, S., Wallin, M.W. 2012. Carrots and Rainbows: Motivation and Social Practice in Open Source Software Development. *MIS Quarterly*. 36, 2 (Jun. 2012), 649–676.

Krogh, G. von, Hippel, E. von 2003. Editorial: Special issue on open source software development. *Research Policy*. 32, 7 (Jul. 2003), 1149–1157.

Krogh, G. von, Spaeth, S., Lakhani, K.R. 2003. Community, joining, and specialization in open source software innovation: A case study. *Research Policy*. 32, 7 (2003), 1217–1241.

Laitenberger, O., Dreyer, H.M. 1998. Evaluating the usefulness and the ease of use of a Web-based inspection data collection tool. *Proceedings of the Fifth International Software Metrics Symposium* (Nov. 1998), 122–132.

Lakhani, K.R., Wolf, R.G. 2005. Perspectives on Free and Open Source Software. *Perspectives on Free and Open Source Software*. The MIT Press. 1–22.

Lampe, C., Johnston, E. 2005. Follow the (Slash) Dot: Effects of Feedback on New Members in an Online Community. *Proceedings of the 2005 International ACM SIGGROUP Conference on Supporting Group Work* (Sanibel Island, Florida, USA, 2005), 11–20.

Lampe, C., Obar, J., Ozkaya, E., Zube, P., Velasquez, A. 2012. Classroom Wikipedia Participation Effects on Future Intentions to Contribute. *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work* (Seattle, Washington, USA, 2012), 403–406.

LaToza, T.D., Towne, W.B., Hoek, A. van der, Herbsleb, J.D. 2013. Crowd development. *Proceedings of the 2013 6th International Workshop on Cooperative and Human Aspects of Software Engineering* (May. 2013), 85–88.

Lave, J., Wenger, E. 1991. *Situated Learning : Legitimate Peripheral Participation*. Cambridge University Press.

Lee, S., Park, D.-H., Han, I. 2014. New members' online socialization in online communities: The effects of content quality and feedback on new members' content-sharing intentions. *Computers in Human Behavior*. 30, (Jan. 2014), 344–354.

Lehman, M.M. 1996. Laws of Software Evolution Revisited. *Proceedings of the 5th European Workshop on Software Process Technology* (London, UK, UK, 1996), 108–124.

Levine, J.M., Moreland, R.L. 1994. Group socialization: Theory and research. *European review of social psychology*. 5, 1 (Mar. 1994), 305–336.

Lofland, J.F., Lejune, R.A. 1960. Initial Interaction of Newcomers in Alcoholics Anonymous: A Field Experiment in Class Symbols and Socialization. *Social Problems*. 8, (1960), 102–111.

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

Midha, V., Palvia, P., Singh, R., Kshetri, N. 2010. Improving open source software maintenance. *Journal of Computer Information Systems*. 50, 3 (2010), 81–90.

Morgan, J.T., Bouterse, S., Walls, H., Stierch, S. 2013. Tea and Sympathy: Crafting Positive New User Experiences on Wikipedia. *Proceedings of the 2013 Conference on Computer Supported Cooperative Work* (San Antonio, Texas, USA, 2013), 839–848.

Musicant, D.R., Ren, Y., Johnson, J.A., Riedl, J. 2011. Mentoring in Wikipedia: A Clash of Cultures. *Proceedings of the 7th International Symposium on Wikis and Open Collaboration* (Mountain View, California, 2011), 173–182.

Nakagawa, E.Y., Sousa, E.P.M. de, Brito Murata, K. de, Faria Andery, G. de, Morelli, L.B., Maldonado, J.C. 2008. Software Architecture Relevance in Open Source Software Evolution:

A Case Study. *Proceedings of the 32nd Annual IEEE International Computer Software and Applications* (Jul. 2008), 1234–1239.

Nakakoji, K., Yamamoto, Y., Nishinaka, Y., Kishida, K., Ye, Y. 2002. Evolution Patterns of Open-source Software Systems and Communities. *Proceedings of the International Workshop on Principles of Software Evolution* (Orlando, Florida, 2002), 76–85.

Naur, P. 1983. Psychology of Computer Use. *Psychology of Computer Use*. Academic Press. 159–170.

O'Reilly, T. 2005. What is Web 2.0: Design Patterns and Business Models for the Next Generation of Software.

Opdenakker, R. 2006. Advantages and Disadvantages of Four Interview Techniques in Qualitative Research. *Forum: Qualitative Social Research*. 7, 4 (Sep. 2006), Art. 11.

Oreg, S., Nov, O. 2008. Exploring motivations for contributing to open source initiatives: The roles of contribution context and personal values. *Computers in Human Behavior*. 24, 5 (Sep. 2008), 2055–2073.

Palen, L., Salzman, M. 2002. Voice-mail diary studies for naturalistic data capture under mobile conditions. *Proceedings of the 2002 ACM Conference on Computer Supported Cooperative Work* (2002), 87–95.

Panciera, K., Halfaker, A., Terveen, L. 2009. Wikipedians Are Born, Not Made: A Study of Power Editors on Wikipedia. *Proceedings of the ACM 2009 International Conference on Supporting Group Work* (Sanibel Island, Florida, USA, 2009), 51–60.

Park, Y. 2008. *Supporting the learning process of open source novices : an evaluation of code and project history visualization tools*. School of Electrical Engineering and Computer Science - Oregon State University.

Park, Y., Jensen, C. 2009. Beyond pretty pictures: Examining the benefits of code visualization for open source newcomers. *Proceedings of the 5th IEEE International Workshop on Visualizing Software for Understanding and Analysis* (Sep. 2009), 3–10.

Pham, R., Singer, L., Liskin, O., Filho, F.F., Schneider, K. 2013. Creating a Shared Understanding of Testing Culture on a Social Coding Site. *Proceedings of the 2013 International Conference on Software Engineering* (San Francisco, CA, USA, 2013), 112–121.

Preece, J. 2001. Sociability and usability in online communities: Determining and measuring success. *Behaviour and Information Technology*. 20, 5 (2001), 347–356.

Preece, J. 2004. Etiquette Online: From Nice to Necessary. *Communications of the ACM*. 47, 4 (Apr. 2004), 56–61.

Qureshi, I., Fang, Y. 2011. Socialization in Open Source Software Projects: A Growth Mixture Modeling Approach. *Organizational Research Methods*. 14, 1 (Jan. 2011), 208–238.

Raymond, E.S. 1999. *The Cathedral and the Bazaar*. O'Reilly & Associates, Inc.

Roberts, J.A., Hann, I.-H., Slaughter, S.A. 2006. Understanding the Motivations, Participation, and Performance of Open Source Software Developers: A Longitudinal Study of the Apache Projects. *Management Science*. 52, 7 (Jul. 2006), 984–999.

Runeson, P. 2003. Using students as experiment subjects–an analysis on graduate and freshmen student data. *Proceedings of the 7th International Conference on Empirical Assessment in Software Engineering* (2003), 95–102.

Salviulo, F., Scanniello, G. 2014. Dealing with Identifiers and Comments in Source Code Comprehension and Maintenance: Results from an Ethnographically-informed Study with Students and Professionals. *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering* (London, England, 2014), 48:1–48:10.

Sande, C. van de 2013. Experiences of a Newbie Helper in a Free, Open, Online, Mathematics Help Forum Community. *Journal of Education and Training Studies*. 1, 1 (Apr. 2013), 194–203.

Santos, C., Kuk, G., Kon, F., Pearson, J. 2013. The Attraction of Contributors in Free and Open Source Software Projects. *The Journal of Strategic Information Systems*. 22, 1 (Mar. 2013), 26–45.

Santos, C.D., Cavalca, M.B., Kon, F., Singer, J., Ritter, V., Regina, D., Tsujimoto, T. 2011. Intellectual Property Policy and Attractiveness: A Longitudinal Study of Free and Open Source Software Projects. *Proceedings of the ACM 2011 Conference on Computer Supported Cooperative Work* (Hangzhou, China, 2011), 705–708.

Scacchi, W. 2002. Understanding the requirements for developing open source software systems. *IEE Proceedings Software*. 149, 1 (Feb. 2002), 24–39.

Schilling, A., Laumer, S., Weitzel, T. 2012. Who Will Remain? An Evaluation of Actual Person-Job and Person-Team Fit to Predict Developer Retention in FLOSS Projects. *Proceedings of the 2012 45th Hawaii International Conference on System Sciences* (Washington, DC, USA, 2012), 3446–3455.

Schweik, C.M., English, R.C., Kitsing, M., Haire, S. 2008. Brooks' Versus Linus' Law: An Empirical Test of Open Source Projects. *Proceedings of the 2008 International Conference on Digital Government Research* (Montreal, Canada, 2008), 423–424.

Seaman, C.B. 1999. Qualitative methods in empirical studies of software engineering. *IEEE Transactions on Software Engineering*. 25, 4 (Jul. 1999), 557–572.

Shah, S.K. 2006. Motivation, Governance, and the Viability of Hybrid Forms in Open Source Software Development. *Management Science*. 52, 7 (Jul. 2006), 1000–1014.

Sharp, H., Baddoo, N., Beecham, S., Hall, T., Robinson, H. 2009. Models of motivation in software engineering. *Information and Software Technology*. 51, 1 (Jan. 2009), 219–233.

Sim, S.E., Holt, R.C. 1998. The ramp-up problem in software projects: a case study of how software immigrants naturalize. *Proceedings of the 20th International Conference on Software Engineering* (Apr. 1998), 361–370.

Singh, V. 2012. Newcomer integration and learning in technical support communities for open source software. *Proceedings of the 17th ACM International Conference on Supporting Group Work* (New York, NY, USA, 2012), 65–74.

Sinha, V.S., Mani, S., Sinha, S. 2011. Entering the circle of trust: developer initiation as committers in open-source projects. *Proceedings of the 8th Working Conference on Mining Software Repositories* (Waikiki, Honolulu, HI, USA, 2011), 133–142.

Smith, S.A., Kass, S.J., Rotunda, R.J., Schneider, S.K. 2006. If at first you don't succeed: Effects of failure on general and task-specific self-efficacy and performance. *North American Journal of Psychology*. 8, 1 (Apr. 2006), 171–182.

Smolander, K., Rossi, M., Purao, S. 2008. Software architectures: Blueprint, literature, language or decision? *European Journal of Information Systems*. 17, 6 (2008), 575–588.

Steinmacher, I., Chaves, A.P., Conte, T., Gerosa, M.A. 2014a. Preliminary empirical identification of barriers faced by newcomers to Open Source Software projects. *Proceedings of the 28th Brazilian Symposium on Software Engineering* (2014), 1–10.

Steinmacher, I., Chaves, A.P., Gerosa, M.A. 2010. Awareness support in global software development: a systematic review based on the 3C collaboration model. *Proceedings of the 16th international conference on Collaboration and technology* (Berlin, 2010), 185–201.

Steinmacher, I., Chaves, A.P., Gerosa, M.A. 2013a. Awareness Support in Distributed Software Development: A Systematic Review and Mapping of the Literature. *Computer Supported Cooperative Work (CSCW)*. 22, 2-3 (2013), 113–158.

Steinmacher, I., Conte, T., Gerosa, M.A. 2015a. Understanding and Supporting the Choice of an Appropriate Task to Start With In Open Source Software Communities. *Proceedings of the 48th Hawaiian International Conference in Software Systems* (2015), 1–10.

Steinmacher, I., Conte, T., Gerosa, M.A., Redmiles, D.F. 2015b. Social Barriers Faced by Newcomers Placing Their First Contribution in Open Source Software Projects. *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing* (Vancouver, BC, Canada, Feb. 2015), 1–13.

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

Steinmacher, I., Silva, M.A.G., Gerosa, M.A. 2014c. Barriers Faced by Newcomers to Open Source Projects: A Systematic Review. *Open Source Software: Mobile Open Source Technologies*. Springer. 153–163.

Steinmacher, I., Silva, M.A.G., Gerosa, M.A., Redmiles, D.F. 2015c. A systematic literature review on the barriers faced by newcomers to open source software projects. *Information and Software Technology*. 59, (Mar. 2015), 67–85.

Steinmacher, I., Wiese, I.S., Chaves, A.P., Gerosa, M.A. 2012a. Newcomers Withdrawal in Open Source Software Projects: Analysis of Hadoop Common Project. *Proceedings of the 2012 Brazilian Symposium on Collaborative Systems* (Oct. 2012), 65–74.

Steinmacher, I., Wiese, I.S., Chaves, A.P., Gerosa, M.A. 2013b. Why do newcomers abandon open source software projects? *Proceedings of the 2013 6th International Workshop on Cooperative and Human Aspects of Software Engineering* (2013), 25–32.

Steinmacher, I., Wiese, I.S., Conte, T., Gerosa, M.A., Redmiles, D. 2014d. The Hard Life of Open Source Software Project Newcomers. *Proceedings of the International Workshop on Cooperative and Human Aspects of Software Engineering* (2014), 72–78.

Steinmacher, I., Wiese, I.S., Gerosa, M.A. 2012b. Recommending mentors to software project newcomers. *Proceedings of the Third International Workshop on Recommendation Systems for Software Engineering* (Washington, DC, USA, Jun. 2012), 63–67.

Stewart, K.J., Ammeter, A.P., Maruping, L.M. 2006. Impacts of License Choice and Organizational Sponsorship on User Interest and Development Activity in Open Source Software Projects. *Information Systems Research*. 17, 2 (Jun. 2006), 126–144.

Stol, K.-J., Avgeriou, P., Babar, M.A. 2010. Identifying architectural patterns used in open source software: approaches and challenges. *Proceedings of the 14th International conference on Evaluation and Assessment in Software Engineering* (UK, 2010), 91–100.

Strauss, A., Corbin, J.M. 2007. *Basics of Qualitative Research : Techniques and Procedures for Developing Grounded Theory*. SAGE Publications.

Suh, B., Convertino, G., Chi, E.H., Pirolli, P. 2009. The Singularity is Not Near: Slowing Growth of Wikipedia. *Proceedings of the 5th International Symposium on Wikis and Open Collaboration* (Orlando, Florida, 2009), 8:1–8:10.

Symon, G. 2004. Qualitative research diaries. *Essential Guide to Qualitative Methods in Organizational Research*. SAGE publications. 98–113.

The Free Dictionary 2014. .

Thoma, L., Lindemann, E. 1961. Newcomers' Problems in a Suburban Community. *Journal of the American Institute of Planners*. 27, 3 (1961), 185–198.

Tidwell, L.C., Walther, J.B. 2002. Computer-Mediated Communication Effects on Disclosure, Impressions, and Interpersonal Evaluations: Getting to Know One Another a Bit at a Time. *Human Communication Research*. 28, 3 (2002), 317–348.

Treude, C., Storey, M.-A. 2010. Awareness 2.0: staying aware of projects, developers and tasks using dashboards and feeds. *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering* (Cape Town, South Africa, 2010), 365–374.

Tsai, H.-T., Pai, P. 2014. Why do newcomers participate in virtual communities? An integration of self-determination and relationship management theories. *Decision Support Systems*. 57, (Jan. 2014), 178–187.

Tsay, J., Dabbish, L., Herbsleb, J. 2014. Influence of social and technical factors for evaluating contribution in GitHub. *Proceedings of the 36th International Conference on Software Engineering* (2014), 356–366.

Van Maanen, J.E., Schein, E.H. 1977. *Toward a theory of organizational socialization*. Technical Report #960-77. Massachusetts Institute of Technology (MIT), Sloan School of Management.

Vaz, V., Conte, T., Travassos, G.H. 2013. Empirical Assessments of a tool to support Web usability inspection. *CLEI Electronic Journal*. 16, 3 (Dec. 2013), 16 pp.

Ververs, E., Bommel, R. van, Jansen, S. 2011. Influences on Developer Participation in the Debian Software Ecosystem. *Proceedings of the Internetional Conference on Management of Emergent Digital EcoSystems* (San Francisco, California, 2011), 89–93.

Viana, D., Conte, T., Vilela, D., Cleidson R.B. Souza, de, Santos, G., Prikladnicki, R. 2012. The influence of human aspects on software process improvement: Qualitative research findings and comparison to previous studies. *Proceedings of the 16th International Conference on Evaluation Assessment in Software Engineering* (2012), 121–125.

Vora, P., Komura, N. 2010. The n00b Wikipedia Editing Experience. *Proceedings of the 6th International Symposium on Wikis and Open Collaboration* (2010), Article 36.

Wang, J., Sarma, A. 2011. Which bug should I fix: helping new developers onboard a new project. *Proceedings of the 4th International Workshop on Cooperative and Human Aspects of Software Engineering* (Waikiki, Honolulu, HI, USA, 2011), 76–79.

Wang, L.S., Chen, J., Ren, Y., Riedl, J. 2012. Searching for the Goldilocks Zone: Trade-offs in Managing Online Volunteer Groups. *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work* (Seattle, Washington, USA, 2012), 989–998.

Wolff-Marting, V., Hannebauer, C., Gruhn, V. 2013. Patterns for tearing down contribution barriers to FLOSS projects. *Proceedings of the 12th International Conference on Intelligent Software Methodologies, Tools and Techniques* (2013), 9–14.

Ye, Y., 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.

Yin, R.K. 2008. *Case study research: Design and methods*. Sage publications.

Yu, S., Ming, W. 2009. Research on individual motivation model of software engineering. *Journal of Communication and Computer*. 6, 11 (2009), 12.

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

Zhou, M., Mockus, A. 2012. What make long term contributors: Willingness and opportunity in OSS community. *Proceedings of the 34th International Conference on Software Engineering* (Jun. 2012), 518–528.

Zhou, M., Mockus, A. 2015. Who Will Stay in the FLOSS Community? Modelling Participant's Initial Behaviour. *IEEE Transactions on Software Engineering*. 41, 1 (2015), 82–99.

Zhu, H., Kraut, R., Kittur, A. 2012. Effectiveness of Shared Leadership in Online Communities. *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work* (Seattle, Washington, USA, 2012), 407–416.

Zhu, H., Zhang, A., He, J., Kraut, R.E., Kittur, A. 2013. Effects of Peer Feedback on Contribution: A Field Experiment in Wikipedia. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France, 2013), 2253–2262.