Barriers faced by newcomers to open source projects: a systematic review

Igor Steinmacher[1], Marco A. Graciotto Silva[1], and Marco A. Gerosa[2]

[1] DACOM – UTFPR Campo Mourao – PR – Brazil
{igorfs, magsilva}@utfpr.edu.br
[2] IME – USP Sao Paulo – SP - Brazil
gerosa@ime.usp.br

Abstract. To remain sustainable, some open source projects need a constant influx of new volunteers, or newcomers. However, the newcomers face different kinds of problems when onboarding to a project. In this paper we present the results of a systematic literature review aiming at identifying the barriers that a newcomer can face when contributing to an Open Source Software project. We identified and analyzed 21 studies that evidence this kind of problem. As a result we provide a hierarchical model that relies on five categories of barriers: finding a way to start, social interactions, code issues, documentation problems and newcomers' knowledge. The most evidenced barriers are newcomers' previous technical skills, receiving response from community, centrality of social contacts, and finding the appropriate way to start contributing. This classification provides a baseline for further researches related to newcomers onboarding.

1 Introduction

Some open source software (OSS) communities composed of volunteers from different parts of the globe contributing and collaborating. According to Qureshi and Fang [14], motivate, engage, and retain new developers is the way to promote a sustainable amount of developers in a project. However, newcomers often face difficulties and obstacles when onboarding to a project [8]. This obstacles can lead newcomers to give up their collaboration. Therefore, a major challenge for OSS projects is to provide ways to support the joining of newcomers.

To reduce these problems, newcomers generally post questions and request help to choose their tasks in forums and mailing list or send emails to developers who have central roles in the project (e.g. owners, project leaders) [13, 22]. However, receiving replies that do not offer guidance or unpolished answers can result in newcomers to give up contributing [18]. Given this scenario, it is important to understand the OSS newcomers needs. This understanding may enable the creation of mechanisms and tools to offer a better support for newcomers.

The objective of this research is to identify the barriers faced by newcomers when onboarding to OSS projects. Onboarding is the stage in which an outsider decides to contribute to a project. Onboarding is highly impacted by a steep learning curve as well as reception and expectation breakdowns [17].

In this paper, the methodology chosen to collect these issues is the Systematic Literature Review. From the best of our knowledge, there is no study that directly focused on problems or barriers encountered by newcomers of Open Source Software projects. On the other hand, several articles report these barriers as a side product of the studies. Thus, knowledge is spread across the literature. This study main contribution is aggregating the barriers evidenced by different studies and creating a model with them.

2 Research Method

To perform our systematic review, we defined the following question: What are the barriers that influence newcomers' onboarding to OSS projects? By answering this question, we aim to capture the barriers that a newcomer can face when contributing to an OSS project. We are not interested in newcomers' motivation to join a project, but in the issues they can face after deciding to contribute.

After using different synonyms and combinations to refine our search, the query presented below was used to retrieve the studies from the following digital libraries: ACM, IEEE, Scopus and Springer Link. These libraries were selected because they index the most relevant venues of computer sciences, mostly written on English, they support searching using boolean expression and provide access to the complete text of the paper. We also consulted specialists for conferences, workshops, journals, and websites that could provide relevant studies for our research. However, no new source was added after their advices.

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

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

We considered for selection the papers that were available for download, written in English, that dealt with newcomers onboarding in open source software projects, that presented experimental results, and that were published in journals or workshop/conference proceedings.

Subsequent to the definition of the primary studies list, the researchers read the full documents. To classify the barriers we followed an “inductive coding” approach [21], which is widely applied in qualitative studies of different knowledge areas. In this kind of approach, the evaluator identifies text segments that contain meaningful units and creates a label for a new category to which the text segment is assigned. Afterwards, connections between the codes are identified and they are grouped according to their properties to represent categories.

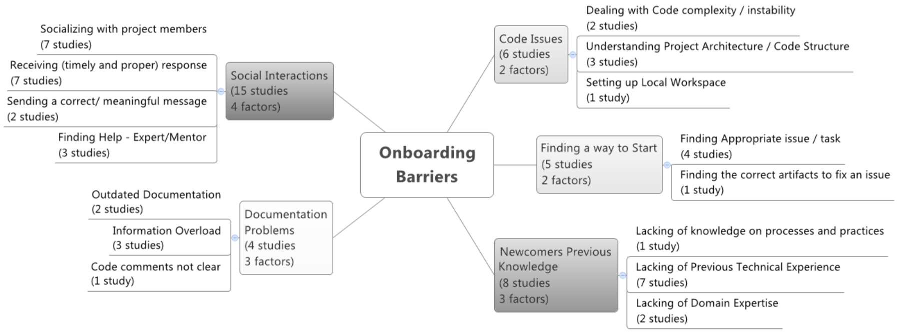
The results of the selection and screening are as follows. After running the query on the digital libraries systems, we got 291 candidate papers. For each paper, title, abstract and keywords were analyzed by two independent researchers.

In a consensus meeting, we came to 33 candidate papers. We checked other papers published by the authors of these 33 candidate studies, finding 20 other candidate papers. After analyzing the abstract of these papers we selected 9 relevant papers, coming to a total of 42 candidate papers. After further analysis, a total of 21 papers were considered relevant to this review and were considered to extract relevant data.

3 Barriers faced by newcomers

The main purpose of this systematic review was to find what are the barriers faced by newcomers to open source projects reported by the literature. For each selected study, we analyzed any barrier reported that was empirically identified or evaluated. We extracted the barriers from the selected studies, and organized them as a hierarchy of barriers, as shown in Figure 3. The figure presents five categories: Social Interactions, Finding a Way to Start, Documentation Problems, Code Issues, and Newcomers' Knowledge.

Fig. 1. Hierarchical map of barriers found in the literature.


Onboarding Barriers
Social Interactions (15 studies 4 factors)
Socializing with project members (7 studies)
Receiving (timely and proper) response (7 studies)
Sending a correct/ meaningful message (2 studies)
Finding Help - Expert/Mentor (3 studies)
Documentation Problems (4 studies 3 factors)
Outdated Documentation (2 studies)
Information Overload (3 studies)
Code comments not clear (1 study)
Code Issues (6 studies 2 factors)
Dealing with Code complexity / instability (2 studies)
Understanding Project Architecture / Code Structure (3 studies)
Setting up Local Workspace (1 study)
Finding a way to Start (5 studies 2 factors)
Finding Appropriate issue / task (4 studies)
Finding the correct artifacts to fix an issue (1 study)
Newcomers Previous Knowledge (8 studies 3 factors)
Lacking of knowledge on processes and practices (1 study)
Lacking of Previous Technical Experience (7 studies)
Lacking of Domain Expertise (2 studies)

In the Figure 3 it is also possible to observe the number of studies that offer evidences for each barrier. The studies were conducted with different projects and different number of projects analyzed. In the following sections we will discuss the barriers found and the evidences that support the problem.

3.1 Social interactions

This category represents the barriers related to the way newcomers interact with the community, including who are the members they exchange messages, the size of their contact network, how they communicate and how the community communicate with them.

Socializing with project members The study conducted by [9] highlights the influence of social and political organization for newcomers willing to become a core developer. The author analyzes mailing list discussions and conduct an in-depth analysis of the socialization of a successful developer. He emphasizes the need to build an identity in the project: *"what the newcomer has to learn is how to participate and how to build an identity that will help get his* *ideas accepted and integrated."* Other authors also report the importance of socializing with central members. For example, Bird [2] quantitatively analyzed mailing lists and report that *"the social network measure, indegree, . . . had a significant effect on immigration."*

All studies that analyzed the centrality/importance of the contacts found that the closer the newcomer is to the center of the community, the more successful the newcomer is. However, the newcomer usually does not choose who will answer her questions. So, when the most appropriate community members receive the newcomers, the chance of retention is higher.

Receiving (timely and proper) response The answers received from the community play an important role during newcomers onboarding. There are evidences of this barrier in seven studies found in this review. Some of them [11, 24, 22, 19] report only the impact of receiving a (timely) response from the community as a barrier. Other researchers [16, 18, 20] also report the impact of the content of responses (properness).

One of the studies that analyze the impact of the answer contents found that *"almost all non-returning newcomers can be attributed to not receiving a response or receiving a condescending response"* [16]. Regarding the studies that analyze the timely response, [11] analyzed mailing list archives and found that *"nearly 80% of newbie posts received replies, and that receiving timely responses, especially within 48 hours, was positively correlated with future participation."*

We can see that community social skills can influence newcomers' decision to contribute to the project. Generally, newcomers demand attention and friendly hands to start contributing. We understand that core members need to stop their main tasks to receive newcomers with no guarantee that they will contribute. However, a good reception can be crucial to retain more newcomers.

Sending a correct/meaningful message [16] reported a problem related to newcomers' communication behavior. By analyzing the history of a support forum they found that *"the newcomers who used informative subject lines for their first message improved chances of getting responses as well as getting their problems solved by the community . . . if the newcomer does not post comprehensible messages or uses a language that the forum responders do not understand. . . ."* Therefore, newcomers who want to be welcomed by the community should focus on the quality of their community-oriented initial interactions.

Finding mentorship/expertise Easiness to find an expert or a mentor is also evidenced in some studies [4, 6]. Cubranic et al. [6] report that *"It can be difficult for newcomers to join such groups [OSS projects] because it is hard to obtain effective mentoring."* To alleviate it, Canfora et al. [4] proposed a tool that recommends mentors to newcomers. They evaluated the tool by surveying some project members and found that mentoring is important to newcomers.

Mentorship is presented as be a good way to help newcomers. However, its actual applicability need to be studied in deep. It is not clear if this kind of policy can be applied in OSS communities, as it depends on experienced volunteers to do this specific task.

3.2 Finding a Way to Start

This category represents the barriers related to difficulties that newcomers face when trying to find the right place to start contributing.

Finding appropriate issue/task Finding the appropriate task to work on was classified as a barrier. Park and Jensen [13] reported that *"...subjects expressed a need for information specific to newcomers, for instance, how to get involved and become active (e.g. communication channels, available sources of information for starters, etc.), what to contribute to (e.g. open issues, required features, sample tasks to start with), and working practices."*

Von Krogh et al. [22] also report on this issue. They found that *"in 56.7% of the cases members of the community encouraged the new participants to find some part of the software architecture to work on that would match with their specialized knowledge. In only 16.7% of the cases new participants were both encouraged to join and given specific technical tasks to work on."* This occurs because, according to their interviews, the community expects new participants to find their own task to work on instead of receiving a specific piece of work.

Communities point of view is that newcomers should be able to find the most appropriate task themselves, as reported by [22]. However, other researches show that the projects should give special attention to this issue [1, 5, 13].

Finding the correct artifacts to fix an issue When the newcomer find a task to work on, another issue can impact his contribution: how to find the correct artifacts. Cubranic et al. [6] proposed Hipikat, a tool that recommends artifacts that are relevant to a task that a newcomer is trying to perform. When conducting an experiment with Eclipse project, they found that *"newcomers can use the information presented by Hipikat to achieve results comparable in quality and correctness to those of more experienced members of the team."*

Newcomers really need support on finding code artifacts related to the chosen task, as projects' structure/architecture are not always trivial and straightforward. So, projects would benefit from tools like Hipikat to support newcomers first steps, as evidenced by the study conducted by Cubranic et al. [6].

3.3 Code Issues

This category comprises the barriers that are related to the source code of the products. To contribute a newcomer usually needs to change existing source code. Therefore, it is necessary for the newcomers to have enough knowledge about the code to start their contributions.

Dealing with Code Complexity/Instability Some studies focus on how code complexity can affect the newcomers to OSS projects. Studying SourceForge projects Midha et al. [12] show that *"cognitive [code] complexity has a strong negative influence on the number of contributions from new developers."* Stol et al. [20] highlight some complaints of newcomers about project structure/architecture of Open Source projects.

Understanding the Project structure/architecture. Stol et al. [20] highlight some complaints of newcomers about project structure of OSS projects. One subject reported that *"the hierarchy of the source code directory was counter intuitive for someone with little architecting experience."* Cubranic and Murphy [7] also present an issue faced during their experiment: *"We also had reports of a pair missing a relevant suggestion because they lacked knowledge about the overall structure of the system..."*

Park and Jensen [13] analyzed *"the potential benefits of information visualization in supporting newcomers through a controlled experiment."* They reported that *"providing visual information such as the class diagrams or dependency views . . . would help new developers understand the structure of existing code and find problems to work on."*

The main complain regarding code is that its structure is hard to understand, and learning it would take too much time. The use of visualization [13], or even artifact recommendation tools [6] can alleviate this problem.

Setting up Local Workspace The feedback obtained by Stol et al. [20] evidenced that newcomers have difficulties when setting up their environment. They reported some obstacles, for example: *"a challenge was that some [subjects] did not have any experience or knowledge on checking out source code from the version control system."* To welcome newcomers, the communities should provide easy access to tutorials and step-by-step cookbooks on how to obtain the code, setup up and build a local workspace.

3.4 Documentation Problems

Project documentation was also explored by some studies. Newcomers need to learn technical and social aspects of the project to contribute. Thus, problems related to documentation were recurrently reported.

Outdated Documentation Steinmacher et al [19] report some issues faced by newcomers regarding outdated information: *"We can see many demotivating facts that occurred in this case: ... outdated information in the issue tracker made the developers waste time on an already existent feature and on checking each issue they pick to address..."* Stol et al.[20] also report issues regarding outdated documentation. They report that the subjects *"... were uncertain whether the available diagrams were still up to date and relevant for the current version of the software... Another reported challenge was the uncertainty whether the available documentation was up to date for the current version of the software."* Finding outdated documentation can make the newcomer gave up onboarding. So, documentation provided by projects must be up-to-date enough to support new developers.

Information overload [13, 7] conduct experiments to assess the benefits that tools that support dealing with information overload can bring to newcomers. Cubranic et al.[7, 6] presents a tool called Hipikat that aims at recommending source code artifacts that should be related to the issue a newcomer is working on. Park and Jensen [13] evaluate the use visualization tools to alleviate problems with overload and report that *"[the tools] provided more efficient ways to handle large amounts of data and understand dependencies in source code, reducing the learning curve and information overload experienced..."* A rich documentation is essential for newcomers trying to understand the projects. However, just providing a bunch of documentation leads to information overload. So, the project should provide easy ways to find this documentation.

Code Comments not Clear In addition to outdated information and information overload, Stol et al. [20] report a problem related code comments. *"It was reported that the code was not very well documented, which made it more difficult to understand the source code."*

3.5 Newcomers' Previous Experience

This category comprises the barriers related to the experience of the newcomers regarding the project and the way they show this experience when joining the projects.

Lacking of Process and Practices We found just one study presenting evidence of learning project practices as a barriers that can hinder the newcomers onboarding. The study conducted by Schilling et al. [15] found that previous knowledge regarding the project practices influence newcomer first steps. They report that *"familiarity with the coordination practices of the project team has a strong association with the time they spend on their projects after GSoC."*

Lacking of Domain expertise Von Krogh et al. [22] claim that *" feature gifts by newcomers emerge from the newcomers prior domain knowledge and user experience."* In the study conducted by Stol et al. [20], the subjects *"reported their unfamiliarity with the domain to be a hindrance."* So, newcomers who present previous domain knowledge have more chances to have a successful onboarding and to be well received by the community.

Lacking of Technical expertise Schilling et al. [15] reported that *". . . level of practical development experience is strongly associated with their continued permanence."* Some studies report sending messages or patches to mailing list or issue tracker presenting previous technical skills can benefit the newcomer when joining. Stol et al. [20] evinced that *"when newcomers mentioned that they had already tried some options to fix their problem and have put efforts to look for a solution in the forums . . . then the responders were quick to respond and were very helpful."*

Ducheneaut [9] reports that *"expertise is not enough to become a core member in Python: one also has to create material artifacts. . . "* Bird et al. [2] also investigate the impact of sending patches when start the contribution and found that *"demonstrated skill level via patch submission plays an important role in Python and Postgres."* All studies evidence that the newcomer who wish to contribute must check if the technical skills required for a given task or project match with their skills. Newcomers can be proactive and search for the required background, but the project must also provide ways for a newcomer to search which tasks fit to his technical profile.

3.6 Summary

Considering the model defined in the Figure 3, based upon barriers identified by using inductive coding through out the selected studies, we can summarize the evidences for each barrier as shown in Table 3.6. The category more throughly studied is social interaction, accounting for 13 studies while the others range from 8 to 9 related studies each.

Due to the nature of the approach to establish the model, there is at least one paper associated with a barrier. Considering the most studied one, we found that the most evidenced barriers are newcomers' previous technical experience, from Previous Experience category, and aspects regarding social network characteristics and response reception, from Social Interaction category.

Table 1. Studies that evidence each barrier

Category	Barrier	Studies
Finding a Way to Start	Finding appropriate task/issue	[1, 5, 13, 22]
	Finding the correct artifacts to fix an issue	[6]
Newcomers' Previous Knowledge	Lacking of Domain expertise	[22, 20]
	Lacking of Previous Technical Experience	[3, 2, 16, 24, 9, 22, 15]
	Lacking of Knowledge on processes and practices	[15]
Code Issues	Dealing with code complexity/instability	[3, 12]
	Understanding architecture/code structure	[6, 13, 20]
	Setting up Local Workspace	[20]
Documentation Problems	Outdated documentation	[19, 20]
	Code comments not clear	[20]
	Information overload	[6, 13, 20]
Social Interactions	Socializing with project members	[3, 2, 10, 23, 24, 9, 14]
	Receiving (timely and proper) response	[11, 16, 24, 22, 18, 20, 19]
	Sending a correct/meaningful message	[16, 24]
	Finding Help - Mentor/Expert	[4, 6, 19]

4 Threats to Validity

This review may have missed some papers that address barriers encountered by newcomers to OSS projects, since we did not search into every possible source and some relevant papers may not contain the chosen terms. To reduce bias, we contacted some specialists in OSS domain. We adjusted the criteria to cover all relevant papers that were of our knowledge and conducted pilot studies.

Most part of the studies analyzed do not present as main focus analyzing the newcomers needs or the problems they face during their first steps. The papers that aim to analyze newcomers obstacles and problems focus on very specific problems. We know that it would be hard – or even impossible – to identify every problem that can affect newcomers. However, keeping the analysis to just some specific problems restricts the value of the outcomes and their applicability.

The findings of this review may have also been affected as the classification is a human process and it is based on some subjective criteria. In particular, the terms of the area do not have a common definition among all studies. The problems were classified based on inductive coding approach, which also relies on manual classification. To reduce interpretation bias related to human process, this review involved two researchers cross checking each paper for inclusion, and a third researcher responsible for reviewing and discussing the information generated after each step.

5 Conclusion

In this paper we identified 21 studies that evidence barriers that can hinder newcomers' onboarding in OSS projects. We aggregated the barriers evidenced across the literature in a single place. By using an inductive coding approach to organize the barriers, we proposed a model that relies on five categories: finding a way to start, social interactions, code issues, documentation problems and newcomers' knowledge. This model is the main contribution of this systematic review.

As a result of this classification we found that the most evidenced barriers are newcomers' lack of previous technical experience, receiving improper response from community, socializing with project members and finding the appropriate task/issue. This classification provides a baseline for further researches related to newcomers onboarding.

As future work we aim to conduct qualitative studies to confirm the barriers evidenced by the literature. We are conducting some interviews with OSS experienced developers and newcomers to verify what are the main barriers faced by newcomers from their perspective. We plan to refine the classification model based on the results of the interviews. Additionally, based on this model it is possible to propose strategies to offer a better support for newcomers, and study how these mechanisms can benefit newcomers.

Acknowledgements

The authors thank Fundacao Araucaria and CNPq (process 477831/2013-3) for the financial support. Marco Gerosa receives a grant from the CNPq and FAPESP. Igor Steinmacher receives grant from CAPES (BEX 2038-13-7).

References

1. X. Ben, S. Beijun, and Y. Weicheng. Mining developer contribution in open source software using visualization techniques. In *3rd Intl. Conf. on Intelligent System Design and Engineering Applications (ISDEA)*, pages 934–937, 2013.
2. C. Bird. Sociotechnical coordination and collaboration in open source software. In *2011 27th IEEE Intl. Conf. on Software Maintenance*, pages 568–573. IEEE CS, 2011.
3. C. Bird, A. Gourley, P. Devanbu, A. Swaminathan, and G. Hsu. Open borders? immigration in open source projects. In *4th Intl. Workshop on Mining Software Repositories*, page 6, 2007.
4. G. Canfora, M. Di Penta, R Oliveto, and S. Panichella. Who is going to mentor newcomers in open source projects? In *Proceedings of the ACM SIGSOFT 20th Intl. Symposium on the Foundations of Soft. Eng., FSE 2012*, Cary, NC, 2012.
5. A. Capiluppi and M. Michlmayr. From the cathedral to the bazaar: An empirical study of the lifecycle of volunteer community projects. In J. Feller, B. Fitzgerald, W. Scacchi, and A. Sillitti, editors, *Open Source Development, Adoption and Innovation*, volume 234 of *IFIP*, pages 31–44. Springer Boston, 2007.

6. D. Cubranic, G. Murphy, J. Singer, and K. Booth. Hipikat: a project memory for software development. *Soft. Eng., IEEE Transactions on*, 31(6):446–465, 2005.
7. D. Cubranic and G.C. Murphy. Hipikat: recommending pertinent software development artifacts. In *25th Intl. Conf. on Soft. Eng., 2003*, pages 408–418, 2003.
8. Barthlmy Dagenais, Harold Ossher, Rachel K E Bellamy, Martin P. Robillard, and Jacqueline P. de Vries. Moving into a new software project landscape. In *32nd Intl. Conf. on Soft. Eng.*., volume 1, pages 275–284, 2010.
9. N. Ducheneaut. Socialization in an open source software community: A socio-technical analysis. *Comput. Supported Coop. Work*, 14(4):323–368, aug 2005.
10. P. He, B. Li, and Y. Huang. Applying centrality measures to the behavior analysis of developers in open source software community. In *2nd Intl. Conf. on Cloud and Green Computing (CGC)*, pages 418–423, nov. 2012.
11. C. Jensen, S. King, and V. Kuechler. Joining free/open source software communities: An analysis of newbies' first interactions on project mailing lists. In *44th Intl.i Intl. Conf. on System Sciences (HICSS)*, pages 1–10, jan. 2011.
12. V. Midha, P. Palvia, R. Singh, and N. Kshetri. Improving open source software maintenance. *Journal of Computer Information Systems*, 50(3):81–90, 2010.
13. Y. Park and C. Jensen. Beyond pretty pictures: Examining the benefits of code visualization for open source newcomers. In *5th Intl. Workshop on Visualizing Software for Understanding and Analysis*, pages 3–10, sep. 2009.
14. I. Qureshi and Y. Fang. Socialization in open source software projects: A growth mixture modeling approach. *Org Res. Meth.*, 14(1):208–238, 2011.
15. A. Schilling, S. Laumer, and T. Weitzel. Who will remain? an evaluation of actual person-job and person-team fit to predict developer retention in floss projects. In *45th Intl. Conf. on System Sciences (HICSS)*, pages 3446–3455. IEEE CS, 2012.
16. V. Singh. Newcomer integration and learning in technical support communities for open source software. In *17th Intl. Conf. on Supporting Group Work*, GROUP '12, pages 65–74. ACM, 2012.
17. I. Steinmacher, M.A. Gerosa, and D. Redmiles. Attracting, onboarding, and retaining newcomer developers in open source software projects. In *Workshop: Global Software Development in a CSCW Perspective*, 2014.
18. I. Steinmacher, I. Wiese, A.P. Chaves, and M.A. Gerosa. Why do newcomers abandon open source software projects? In *Intl. Workshop on Coop. and Human Aspects of Soft. Eng. (CHASE)*, jun. 2013.
19. I. Steinmacher, I. Wiese, and M.A. Gerosa. Recommending mentors to software project newcomers. In *3rd Intl. Workshop on Recommendation Systems for Soft. Eng. (RSSE)*, pages 63–67, june 2012.
20. K-J. Stol, P. Avgeriou, and M. Ali Babar. Identifying architectural patterns used in open source software: approaches and challenges. In *14th Intl. Conf. on Evaluation and Assessment in Soft. Eng.*, pages 91–100, Swinton, UK, 2010. BCS.
21. D.R. Thomas. A general inductive approach for analyzing qualitative evaluation data. *American Journal of Evaluation*, 27(2):237–246, 2006.
22. G. Von Krogh, S. Spaeth, and K.R. Lakhani. Community, joining, and specialization in open source software innovation: A case study. *Res Policy*, 32(7):1217–1241, 2003.
23. M. Zhou and A. Mockus. Does the initial environment impact the future of developers. In *33rd Intl. Conf. on Soft. Eng. (ICSE)*, pages 271–280, may 2011.
24. M. Zhou and A. Mockus. What make long term contributors: Willingness and opportunity in oss community. In *34th Intl. Conf. on Soft. Eng. (ICSE)*, pages 518–528, june 2012.