Understanding the Impressions, Motivations, and Barriers of One Time Code Contributors to FLOSS Projects: A Survey

Amanda Lee and Jeffrey C. Carver
Computer Science Department
University of Alabama
Tuscaloosa, AL USA
aslee1@crimson.ua.edu; carver@cs.ua.edu

Amiangshu Bosu
Department of Computer Science
Southern Illinois University
Carbondale, IL USA
abosu@cs.siu.edu

***Abstract*—Successful Free/Libre Open Source Software (FLOSS) projects must attract and retain high-quality talent. Researchers have invested considerable effort in the study of core and peripheral FLOSS developers. To this point, one critical subset of developers that have not been studied are *One-Time code Contributors (OTC)* – those that have had exactly one patch accepted. To understand why OTCs have not contributed another patch and provide guidance to FLOSS projects on retaining OTCs, this study seeks to understand the impressions, motivations, and barriers experienced by OTCs. We conducted an online survey of OTCs from 23 popular FLOSS projects. Based on the 184 responses received, we observed that OTCs generally have positive impressions of their FLOSS project and are driven by a variety of motivations. Most OTCs primarily made contributions to fix bugs that impeded their work and did not plan on becoming long term contributors. Furthermore, OTCs encounter a number of barriers that prevent them from continuing to contribute to the project. Based on our findings, there are some concrete actions FLOSS projects can take to increase the chances of converting OTCs into long-term contributors.**

***Index Terms*—FLOSS, Open source, OSS, Newcomers, One Time Contributors, Survey, Qualitative Research**

I. Introduction

For Free / Libre Open Source Software (FLOSS) projects to survive, they must attract and, more importantly, retain new contributors [8]. Popular FLOSS projects (e.g., Apache, Linux, and Android) are able to attract plenty of motivated volunteers [15], [18], [26]. However, many of the project newcomers lack the necessary domain knowledge and programming skills to author acceptable code changes [23], [26]. Prior research on identifying and removing the barriers faced by newcomers to FLOSS projects found that newcomers require additional assistance and may turn away if ignored or treated poorly [14], [20], [24], [28]. Therefore, members of FLOSS projects should take care to ensure that they provide this additional support to those newcomers who have the greatest potential to join the project and become long-term contributors.

Researchers often characterize the structure of FLOSS development communities as *core and periphery*, with a small number of core developers and a larger number of peripheral developers [9], [19], [30]. The core developers are those who have been involved with the FLOSS project for a relatively long time and make significant contributions to guide the development and evolution of the project [30]. The peripheral developers are those that occasionally contribute to the project (e.g. via bug reports or mailing list participation) [30], mostly fix bugs, and do not contribute much in the way of new features [21]. Successful FLOSS projects must attract and retain these peripheral developers so they can provide more significant code contributions.

We define a ***O**ne-**T**ime code **C**ontributor (**OTC**)* to a FLOSS project as *a peripheral developer who has had exactly one code contribution (i.e. a patch) accepted in that project.* While OTCs might also interact with the project via the mailing lists or by reporting bugs, they have successfully contributed only one patch to the project. Other peripheral developers who make single contributions in the form of mailing list posts or comments in the bug review tool are not OTCs. The fact that OTCs have an accepted code contribution indicates that they have generally invested more effort than other peripheral developers [21].

Prior work in this area has examined the motivations of core developers [1], [18], [22], [30] and of peripheral developers [21]. A recent study by Pinto et al. noted the perceptions and characteristics of the contributions by *casual contributors*, which includes OTCs as well as other peripheral developers [20]. These previous studies have not addressed the motivations of OTCs specifically nor identified the important barriers they face in contributing to FLOSS. FLOSS projects could see great benefit from capitalizing upon the initial connection between OTCs and the project by better understanding OTCs' motivations and removing their barriers. In addition, OTCs can provide insights into why some FLOSS contributors choose to leave a project, even after successfully contributing code to that project. Therefore, the goal of this study is *to understand the impressions, the motivations, and the barriers of OTCs in FLOSS projects.* This understanding can help the leaders of FLOSS projects identify approaches to encourage OTCs to form longer-term connections to the project.

To address this goal, we sent an online survey to 4,127

OTCs gathered via mining the code review repositories of 23 popular FLOSS projects. The survey received 184 responses from OTCs. Multiple coders systematically analyzed the open-ended responses using an Open Coding approach to understand the perceptions of OTCs during their interaction with the FLOSS project. The coding process consistently achieved high inter-rater reliability.

The primary contributions of this study are:

- A better understanding of OTCs' impressions, motivations, and barriers.
- Identification of a particular set of FLOSS participants that require more attention.
- Concrete recommendations to FLOSS projects regarding how to retain OTCs.

The rest of the paper is organized as follows. Section II describes the research questions. Section III describes the research methodology. Section IV characterizes the study participants. Section V provides the results. Section VI discusses the implications of the results. Section VII addresses the threats to validity. Finally, Section VIII concludes the paper.

II. Research Questions

This section discusses related work and expands the overall research goal into four more specific research questions which, in turn, drive the survey design.

A. Impressions

While FLOSS peripheral and core developers have different roles and functions, they frequently interact with each other, though usually virtually. Specifically, peripheral and core developers often have a high degree of communication [1]. These interactions provide the opportunity for these developers to form impressions of each other. A common type of interaction is for peripheral developers to identify bugs that core developers then fix [21]. The relationship between core and peripheral developers is a symbiotic one [22]. Core developers depend on peripheral developers to find the bugs. Peripheral developers depend on core developers to fix those bugs.

These types of interactions suggest that developers' impressions of each other can be vital to whether a peripheral member continues to contribute to the project. If a peripheral developer dislikes project members or if she is treated rudely by them, she is less likely to remain with the project [3], [24]. Moreover, a good impression of a project can motivate a potential contributor to make her first contribution [26]. Extrapolating these observations to OTCs suggests that OTCs' impressions may be a key factor in continued participation. To better understand the types of impressions OTCs have of their project, the first research question is:

RQ1: What kind of impressions do One-Time Contributors form about their fellow contributors?

B. Motivation

Various FLOSS contributors may have different motivations for participating in a FLOSS project. Understanding these motivations can provide additional insights to FLOSS projects that wish to attract additional participants. One way to categorize the motivations is **need-based** – *need code for work or home or need to give back to the community* vs. **hobbyists** [22]. A slight expansion of this dichotomy divides **need-based** contributors into those whose needs are **employment-based** – *need the code for work or are paid to contribute*, **need-based** – *use the software personally*, or **reciprocation-based** – *have a desire to give back to the community* [18]. Those who are **hobbyists** may be motivated either by the desire to improve their skills [18] or because they are already quite skilled and enjoy participating in such projects [22]. While peripheral developers can be hobbyists, peripheral developers are usually need-based contributors [21].

Another perspective from which researchers view motivation is **intrinsic** – *driven to contribute by something internal* vs. **extrinsic** – *driven to contribute by something external*. Most peripheral developers are extrinsically motivated. Specific extrinsic motivations include personal or work use, being paid by an employer, an increase in personal reputation, or the desire for a new feature. The lone intrinsic motivation common among peripheral developers is a sense of reciprocity, which is either a desire to give back and foster future contributors or a perceived social obligation to the project [18], [21], [22].

Therefore, OTCs might be expected to share attributes with other peripheral, need-based contributors, and perhaps be motivated by the same goals, such as fixing bugs or enhancing personal reputation. With a greater understanding of OTCs' motivations, FLOSS communities will be better informed on how to retain them, and potentially other peripheral developers, in the project longer. This observation leads to the second research question:

RQ2: What motivations drive One-Time Contributors to contribute?

C. Barriers

The process of joining a FLOSS project can be daunting. Developers have to understand the submission process, interpret semi-automated rejection messages, and handle other difficulties. Researchers have studied this process to identify the barriers commonly faced by individuals who wish to become project members. Steinmacher et al. identified 48 barriers faced by developers when making their first contribution to a FLOSS project. These barriers include: being ignored, trouble deciphering the source code, finding a bug, and even the process of submitting the contribution [24]. Several of these contribution barriers were corroborated in other studies, such as being ignored [17], finding a bug [27], and developing tests for the patch [29].

Difficulties can also arise from from dealing with the project's often proprietary environment and a lack of knowledge of the project's programming language [14], [26], [29]. The project's infrastructure can also pose a barrier. Newcomers will not have much knowledge of how the project is set up, or might choose to fix what turns out to be a larger or

more difficult issue than they anticipated. Both the process of coding a patch and the process of submitting a patch can pose barriers [26].

In addition, projects typically do not make these barriers known to potential contributors. Specifically, newcomers have to learn the social structure of the project and the organization of the codebase on their own [10].

We can consider an OTC to be a successful newcomer because they have already had a code contribution accepted into the project's codebase. These OTCs likely faced some of the same barriers described above, but somehow overcame them to be successful. How the OTCs view these barriers is important information that can help FLOSS projects who wish to convert OTCs into longer-term contributors. Therefore, the third research question is:

RQ3: What barriers do One-Time Contributors face which prevent them from continuing to contribute?

Conversely, there may be times when an OTC's lack of further contribution is not due to any specific barriers. Zhou and Mockus found that a newcomer's personal motivation significantly affected the chance that s/he stayed with the project for many patches [31]. The natural cycle for most FLOSS developers, whether core or periphery, is to leave the project within a year of joining [22]. It may be that OTCs simply leave the project earlier than other developers. If OTCs are not motivated to submit additional patches, or find it natural to leave after a single patch, there is little that projects can do to retain these developers. Projects should not focus any additional resources on attempting to recruit them. Therefore, the fourth research question is:

RQ4: If no barriers stand in the way of One-Time Contributors, why do they choose to leave after a single patch?

III. Methodology

Based on the four research questions, which are geared towards gathering the opinions of members of a population, and the fact that there is little prior research about OTCs, we chose a survey as our research approach. Using the research questions and prior results about peripheral developers in general as a guide, we constructed a survey containing qualitative and quantitative questions. The remainder of this section describes the survey design, the participant selection criteria, pilot testing, data collection, and qualitative data analysis.

A. Survey

Our goal in designing the survey was to keep it as short as possible, while still gathering all of the relevant information. For the current paper, we only consider a subset of the survey questions. Table I lists each survey question included in this paper, the research question that motivated its inclusion, and the answer choices provided. Note that questions indicated with a 'D,' rather than a 'RQ#' were included to gather demographics about the respondents. For the questions that were open-ended, there are no specified answer choices. Several of the questions were inspired by previous surveys [3], [24].

B. Participant Selection

In a previous study about peer impressions in code review [6], we mined data from 23 popular FLOSS projects that used the Gerrit code review tool. This dataset was ideal for reuse in the current study for the following reasons:

- The projects were diverse, including participants with varied backgrounds, interests, and goals.
- The projects were all mature, with hundreds of developers and products available for download.
- The projects required every submitted patch to undergo mandatory peer code review, requiring each OTC to interact with project members at least once to get the patch accepted.
- The projects' repositories contain details about each patch, including the email address of the submitter.

We analyzed this dataset to identify participants who had only one successful code contribution to a given project. This analysis identified 4,587 individuals. Following a process similar to Bird et al.'s [2], we manually audited this list to ensure that it contained only people who were truly OTCs. We performed identity matching to eliminate anyone who had multiple accounts and had patches accepted from more than one account. To improve the validity of our sample, if we were uncertain whether a person had multiple accounts, we eliminated them. Our study stood to suffer more from mistakenly including people who were not OTCs than from mistakenly excluding someone who was an OTC. In the rare case where the same individual was an OTC in multiple projects, we kept them in the study because they were still an OTC in each project. This process resulted in 4,127 unique individuals that were candidates for the survey.

C. Pilot Survey

To help ensure the validity of the survey, we asked Computer Science professors and graduate students with experience in FLOSS and experience in survey design to review the survey to ensure the questions were clear and complete. The feedback only suggested minor edits. The changes we made include: changing a question from allowing only a single-answer to allowing multiple answers, changing the granularity of the sliding bars, and clarifying the wording of some questions.

D. Data Collection

We sent each of the 4,127 OTCs in our database a personalized email identifying the project with which the OTC was connected with a link to the survey. Approximately 600 of those emails bounced, leaving at most 3,500 potential participants, assuming all other emails actually reached their intended recipient. From the 350 that started the survey, 187 completed it. We only analyzed the responses from those that completed the survey and answered at least one qualitative

TABLE I
SURVEY QUESTIONS

#	RQ*	Question Text	Answer Choices
Q1	D	Have you contributed to other FLOSS projects?	[yes, no]
Q2	D	How many other FLOSS projects have you contributed to?	[#]
Q3	D	Estimate the total number of patches you have contributed to all FLOSS projects	[#]
Q4	D	What is your highest level of experience in the field of software development/engineering?	[none, hobbyist, student, professional]
		Q5, Q6, and Q7 only appeared based on the answer to Q5	
Q5	D	What type of student are you currently?	[high school; undergrad college, studying CS or related; undergrad college, studying unrelated; grad school, studying CS or related; grad school, studying unrelated; grad school, studying unrelated with undergrad in CS or related]
Q6	D	How many years have you been employed as a software engineer/developer?	[1-3, 4-6, 7-9, 10+]
Q7	D	How would you describe your level of software engineering/development ability?	[high school, took some in college, took some in grad school, retired professional, learned at job, self taught]
Q8	D	Do you interact with the project outside of the code reviews? If so, how?	[mailing list, code review database, forum/message board, rss feed, bug repository, other, I do not interact outside of code reviews]
Q9	RQ1	What was your initial impression of the other project members?	
Q10	RQ1	What did you think of the other project members' code review responses?	[sliding bar, 1-7]
Q11	RQ2	What motivated you to contribute your patch?	
Q12	D	Are you, or were you, being paid to contribute to this project?	[yes, no]
Q13	D	OPTIONAL: Please list the company that paid you to contribute. Feel free to leave blank.	
Q14	RQ3	How difficult was it to create and submit patches, compared to your previous experiences with FLOSS?	[2 sliding bars, 1-7, one labeled creation, one labeled submission]
Q15	RQ3	Was patch creation or submission hard enough you would not do it again?	[yes, no]
Q16	RQ3	Was patch creation or submission the difficult one, or were both? What was most difficult about them?	
Q17	RQ3, RQ4	Were there any barriers or reasons that kept you from continuing to contribute? If so, please list them below.	

**numbers refer to the research question that motivated the inclusion of the survey question, 'D' refers to demographic questions*

question. We had to exclude 13 respondents because they did not qualify as OTCs (i.e. their patch had been rejected instead of accepted), leaving 174 responses for analysis. As the qualitative questions were optional, some respondents did not provide answers to all of those questions. In this case, we were still able to analyze their responses to the qualitative questions they did answer to gain insight into the respective research questions.

E. Qualitative Analysis Process

For the open-ended questions, we did not have an existing set of codes from which to begin, due to the novelty of the topic. To prevent any initial biasing of the results, we used an Open Coding approach to let the coding scheme emerge during the analysis [13], [25]. Because we were using open coding, we chose to perform the coding in three phases, checking for consistency after each phase. We used Nvivo[1] to support the coding process. At each phase, we computed Cohen's kappa [7] to measure the level of inter-rater reliability in the coding process.

First, each author independently coded the first 30 responses to each question. As a result, each author developed his/her own set of codes to describe the themes that emerged. After meeting to discuss the resulting codes and agreeing upon a consistent coding scheme, each author recoded the first 30 responses using those codes. The resulting kappa value was 0.3792. The authors met and resolved all discrepancies and gained a better understanding of the coding scheme. Second, to check the consistency of understanding of the coding scheme, each author independently coded the next 50 responses, only adding new codes when necessary. The resulting kappa for these 50 responses was 0.7669. Again, the authors met to compare results and resolve any discrepancies. Finally, because the coding scheme was now well-understood, each author independently coded the remainder of the responses. The resulting kappa for the remaining responses was 0.7767. The authors met again to resolve any discrepancies. In the end, all authors agreed on all codings[2] Once we completed the coding process, we transferred the data into SPSS[3] for further analysis along with the quantitative data.

While there is no universally accepted 'good' kappa, values of .75 and up are generally recognized as exceptional scores [11], [16]. The first kappa was low because the two coders were still finalizing the coding scheme. Conversely, the last two kappa values, which cover the bulk of the data, fell into the exceptional range. This high level of inter-rater reliability result indicates that the coding scheme is valid and consistently describes the data.

IV. DEMOGRAPHICS

To provide context for the results described in Section V, this section characterizes the projects from which the re-

[1]Nvivo http://www.qsrinternational.com/nvivo-product

[2]Full coding scheme: http://carver.cs.ua.edu/Data/2017/ICSE_2017/

[3]http://www.ibm.com/analytics/us/en/technology/spss/spss.html

spondents were drawn, the level of experience of the survey respondents (both in Computer Science, generally, and in FLOSS, specifically), whether the respondents were paid to contribute, and the types of interactions the respondents had with the project.

Among the 23 projects from which we solicited survey responses, including Android, Qt, Cyanogen, Gerrithub, AOKP, Openstack, and Wikimedia, the percentage of OTCs ranges from 0.05% to 4% of the total project contributors. The average across all projects is 0.6%. While this percentage may seem small, due to the size of many of these FLOSS projects even a small percentage can represent several hundred contributors. In fact, one project has over 1000 OTCs.

The majority of the respondents were highly experienced in computer science, with at least some work experience in the field. The distribution of the respondents' experience is:

- **Professionals** - 81%, almost $1/2$ of whom have more than 10 years experience;
- **Students** - 10%, almost 3/4 of whom are studying computer science or a related field;
- **Hobbyists** - 8%;
- **None** - 1%.

Regarding FLOSS experience, the respondents exhibit a wide range of experience. Some have very little experience; others have very extensive experience. Approximately 1/2 have contributed to five or fewer FLOSS projects, with the other half contributing to more than five FLOSS projects. A small number, 6%, contributed to only one FLOSS project. Of those who contributed to other projects,

- 28% contributed 1 - 10 total patches;
- 28% contributed 10 - 100 total patches;
- 24% contributed 100 - 999 patches; and
- 20% contributed $>$ 1000 patches.

In many cases, FLOSS contributors are paid by an employer to make contributions. Often the employer's reliance upon the FLOSS project provides the incentive for them to pay employees to contribute. In our sample, only 26% of the respondents were paid to contribute to their respective FLOSS project. This percentage is lower than observed in other FLOSS studies [3], [18], [22]. One reason for this lower percentage could be that because these contributors are paid by an employer, they are likely to make multiple contributions, so they will not remain OTCs for long.

As all the FLOSS projects we surveyed practice mandatory code review, each of the OTCs interacted with the project via the code review tool, Gerrit. Overall, the respondents had a significantly positive impression of the code review process (one sample t-test $p < .01$). In addition, most respondents, 77%, interacted with the project outside of the code review process. These interactions occurred most commonly in the bug repository or the project mailing list, though they also interacted through forums, message boards, and RSS feeds. Several respondents indicated the use of multiple methods of interactions. Overall, these respondents seem to be well-connected to the projects for which they are OTCs.

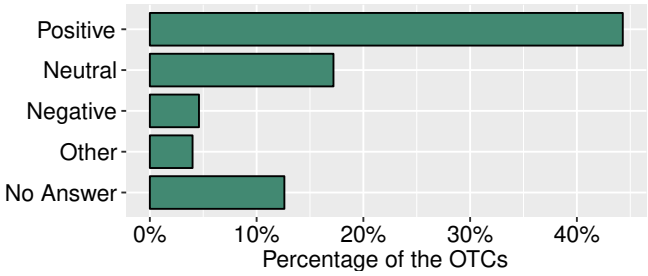

Fig. 1. Overall Impressions of OTCs about Project Members

Overall, these demographics show that the survey respondents were generally experienced in software development, experienced in contributing to FLOSS projects, and active in interacting with the project in addition to patch submission. It is unlikely that a lack of understanding of FLOSS or a lack of development experience led to respondents remaining OTCs. Because they were generally engaged with the projects through additional means of interacting, there must be other reasons why these respondents did not make other successful code contributions. Therefore, this set of respondents should provide valuable insights for the goals of this study.

V. Results

This discussion of the results is organized around the four research questions posed in Section II.

A. RQ1: What kind of impressions do One-Time Contributors have about their fellow contributors?

The answer to this research question came from survey question Q9 (see Table I). Because each response might contain more than one impression, we coded each impression separately. After coding each impression, we assigned each response to one of four high-level categories. *Positive* responses were those where all the impressions in the response were positive. Likewise, *negative* responses were those where all the impressions in the response were negative. *Neutral* responses were those that were either truly neutral or those where there was both a positive and a negative impression in the response. *Other* responses were those that were unintelligible or did not truly answer the question.

As Figure 1 shows, the overall tone of 2/3 of those that responded was positive. This trend suggests that OTCs are not dissuaded from further participation due to a poor impression of the FLOSS project or its members. The remainder of this section describes the results based on the specific impressions given by the respondents, rather than the overall tone. That is, if a *neutral* response has both a positive impression and a negative impression, we considered each impression separately under their respective analyses.

1) Positive Impressions: Respondents overwhelmingly had a positive impression of the FLOSS project and community members. Figure 2 shows the distribution of the impressions among the specific positive responses given. Note that total is

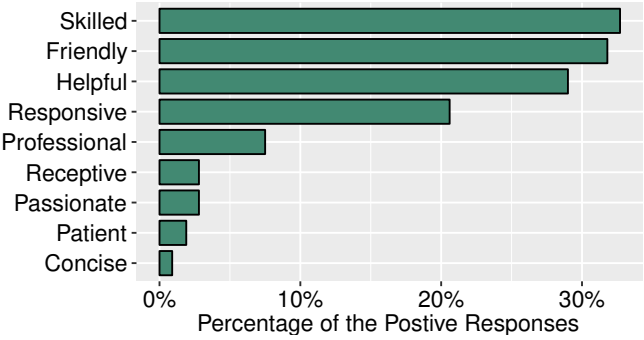

Fig. 2. OTCs' Detailed Positive Impressions of Project Members

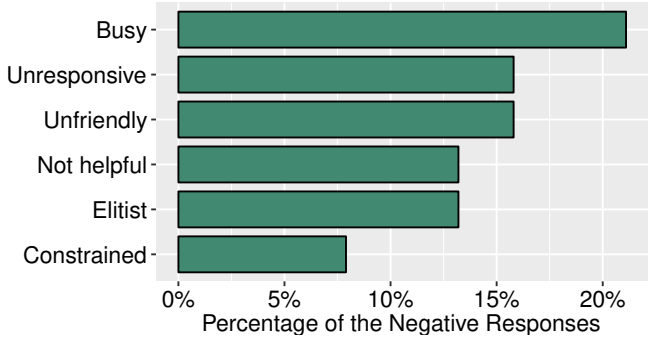

Fig. 3. OTCs' Detailed Negative Impressions of Project Members

greater than 100% respondents because may have listed more than one impression in their answer.

"*Passionate experts; helpful; straight to the point; responsive*," is a representative quote of the positive responses. The positive impressions were very glowing, and often included more than one praise. OTCs with positive impressions were impressed by the **skills** of the project members and considered them as *"... experts on what they do."*.

Just over 1/4 of the OTCs found the project members *"very helpful and friendly."* For example, they were helping the OTCs *''... in polishing the patch and make it conform the project"* or *"... answer even the simplest questions."* Although the project members were willing to assist the OTCs, they remained **professional** during their communications. For example, when accepting patches they were *"... demanding perfection, for the good of the project."* Many of the OTCs with positive impressions also found the project members *responsive* as they were providing *"great response to questions."*

2) Negative Impressions: Accounting for a very small percentage of the responses, Figure 3 shows the distribution of the specific negative impressions identified by the respondents. As opposed to the positive responses, these responses tended to be very negative. The respondents appeared to be greatly displeased with the project or something specific about the project. "*Slow to respond, arrogant, bureaucratic*," as a contributor commented.

In general, the unresponsiveness of the project members was a major source of negative impressions. For example some of the project members *".. did not post timely updates to submitted issues."* However, many of the OTCs also acknowledged that they were **busy**, *"possibly overworked"*, and did not have "*much time to try to include features from outside people*."

A few of the OTCs found the project members to be **unfriendly** or **not helpful** as they *"like to use jargon & killing new ideas"* or *"tend to not let others share their expertise."* As experts in their fields, a few project members became arrogant and were acting "*a little **elitist***."

3) Skill vs. Responsiveness: Combining the top responses for the **positive** impressions with the top responses for the **negative** impressions provides some additional insight. Based on the **positive** responses, OTCs look up to fellow project members because of their perceived skill, friendliness, helpfulness, or responsiveness. From the **negative** responses, OTCs had poor impressions about project members who were too busy to interact with them or were unresponsive. The overall sentiment of these responses is that OTCs have a more positive view of the project when the project members are skilled and pay attention to them.

However, the most common positive impression was **skilled**, which is not related to attentiveness. Half of the respondents who expressed this sentiment were focused purely on the project member's skill level, without any additional comments. Even though they are only contributing to the project once, OTCs value project members whom they perceive as skilled.

Despite this, the responses showed that in some cases skilled project members were less than responsive. A comment from one respondent was, "*highly experienced... not super helpful. But I don't blame them because everyone is busy and they have no obligation to 'mentor' me.*" The responses seemed to indicate some type of inverse relationship between skill and responsiveness, with project members who were more skilled being less likely to be responsive. One respondent commented, "*High level of expertise...not too responsive if their expertise is really high but better if it is not that much*." In other words, OTCs could tell that the project members who were skilled were also difficult to communicate with, at least in part due to their level of skill and tendency to look down on the OTCs. This view was indicated by approximately 25% of the respondents who indicated that the project members were highly skilled.

Conversely, just over 25% of the respondents who indicated that project members were highly skilled also found them to be responsive. This view is characterized by the following comment, "*Extremely talented individuals and highly responsive*." This view indicates that there does not have to be an inverse relationship between skill and responsiveness. Overall, the respondents that held this view were more pleased with their fellow project members, with fewer negative comments in their responses. Therefore, we can conclude that skill alone is not sufficient for imparting a positive impression. Project members also have to be responsive to OTCs if they want to increase the chances of a positive impression.

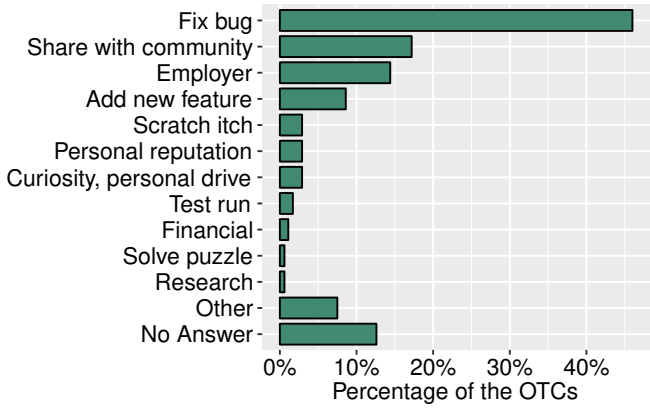

Fig. 4. Motivations Driving OTCs to Submit Code Changes

B. RQ2: What motivations drive One-Time Contributors to contribute?

The answers to this research question came from survey question Q11. The coding of the data resulted in the distribution shown in Figure 4. The reminder of this section examines each of the top four answers in more detail (note that the **Other** category is a catch-all for one-off responses that did not fit in other categories, therefore we do not discuss it in detail).

1) Fix a Bug: The most common motivation for an OTC was the desire to fix a bug that was interfering with their personal needs or their employer's needs. Most respondents had little more to say than, "*Trying to resolve an issue directly affecting me.*" Simply put, because they were bothered by a bug, they fixed it to remove the problem.

Respondents who gave this motivation are potentially less likely to become long-term contributors. To better understand how this motivation might have impacted the overall results, we performed a secondary analysis. We compare the respondents who mentioned the **fix bug** motivation to those respondents who mentioned other motivations.

The relative distribution of motivations after removing **fix bug** is similar to the distribution in Figure 4 (which has all respondents). Despite the fact that respondents could express multiple motivations, the overall distribution does not change when removing the **fix bug** response. This observation suggests that those with the **fix bug** motivation typically had no additional motivation for submitting their patch.

While there was little difference in the frequency of encountering barriers (discussed in Section V-C), we did observe some differences between those with the **fix bug** motivation and the other respondents. Compared with the other respondents, those with the **fix bug** motivation had less experience and communicated through fewer channels.

Overall, OTCs who expressed the **fix bug** motivation appear to be less invested in the project than OTCs who had other motivations. An individual's motivation and level of investment in a project can significantly affect their chances of becoming a core contributor [31], therefore the FLOSS community should focus their efforts on those OTCs whose code contributions were motivated by reasons other than simply fixing a bug.

2) Share with Community: The desire to give back to the community, otherwise known as *reciprocity*, was the second most common motivation for OTCs. These OTCs appear to be more community-minded and contributed code to prevent other users from encountering the same bug they faced. According to one respondent, they contributed "*[t]o better the software, and to benefit other people*." Even though OTCs only successfully contributed code to the project one time, they made the contribution for the good of the project or out of a sense of obligation to the project. That the reciprocal culture in FLOSS projects extends all the way to the OTCs speaks to the strength of the FLOSS culture.

3) Employer: The third most common motivation for OTCs was work or an employer. Rather than being paid by their employer to contribute to the FLOSS project, many OTCs reported that they contributed the patch to the FLOSS project to fix a bug or ease the use of some software or tool they needed for work. Two representative quotes from respondents are: "*[w]e used the software as part of our build process and the patch fixed a bug that was causing us a great deal of trouble,*" and "*[f]ixing a bug that caused problems for my company's products.*"

Other respondents reported that their motivation was to reduce internal maintenance for their employer. Two representative quotes from respondents are: "*[f]ixing them would simplify upmerging*" and "*integrating it into the mainline Android, means less work for us in the future.*" By contributing the patch, the respondent made his own job easier by outsourcing patch maintenance to the FLOSS project.

4) Add New Feature: The fourth most common motivation was a desire to add a new feature to the product. Some OTCs contributed a feature for their own benefit or because they required it themselves. One respondent contributed because of "*[b]ugs I ran into or missing functionality I needed*." Other respondents wanted to help other contributors as well as themselves. According to one respondent, s/he contributed to "*[i]mplement functionality needed by us... beneficial also to others without having to maintain a patch in our downstream fork.*" These respondents were motivated by the desire to have their code included in and maintained by the project, both so that they would not have to personally maintain it and to help out other contributors who might require the same functionality. Note that these responses also were coded into the **share with community** category.

C. RQ3: What barriers do One-Time Contributors face which prevent them from continuing to contribute?

The overall results for this question came from survey Q17. Figure 5 shows the distribution of responses. Just under half of those that responded (47%) indicated that they faced one or more barriers that prevented them from making additional contributions to their respective FLOSS projects. Slightly fewer respondents indicated that they did not face a barrier. The remaining were ambiguous or unintelligible, and therefore could not be placed in either group.

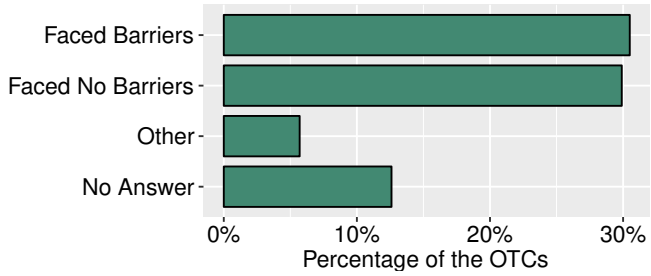

Fig. 5. Whether OTCs Experience Barriers

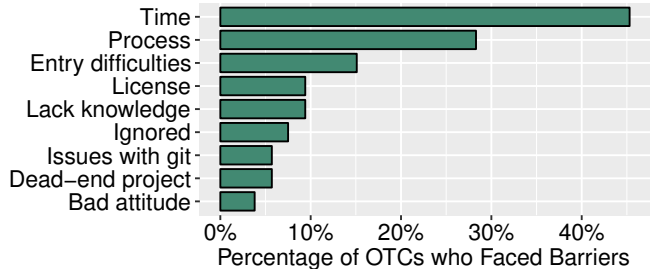

Fig. 6. Specific Barriers Faced by OTCs

If respondents answered 'yes' to survey Q17, s/he had the opportunity to describe the barriers s/he faced. Figure 6 shows the distribution of the responses to this question. The reminder of this section analyzes each of those responses in more detail.

1) Time: The most common barrier faced by OTCs was time. These respondents did not have time to contribute additional code to the project, so they left. Respondents gave various explanations of how the lack of time impacted them. Many respondents did not provide any explanation beyond the simple answer of "*[t]ime.*" Other respondents focused on the length of the review process. One respondent indicated that the "*extremely slow turnaround time on review/resolution of bug reports*" is what prevented them from submitting subsequent code. Other respondents had other duties from their employer with left no time for further code contributions. One respondent indicated that s/he had a "*very busy full time job not related to project.*" Still other respondents indicated a "*lack of time needed to further familiarize myself with the codebase and the project in general.*" While it might seem difficult for FLOSS projects to encourage these respondents make additional code contributions, some of the reasons given for the time barrier could be addressed. For example, projects could strive to provide a quicker turnaround on patch reviews, especially for new contributors.

2) Process: The most common complaint about the process was that the submission process was too long or too complex. In addition, some respondents considered the code review process to be too complex or lengthy, particularly for submitting a simple code contribution. As one respondent put it, "*[l]earning the process seemed like more work than actually creating the patch...*" In these cases, the respondents found the process so arduous and lengthy that it was not worth it to stay with the project and learn the process.

Additional insight provided by the responses to survey questions Q15 showed that a relatively small number of respondents (15%) indicated that the patch creation or patch submission process was difficult enough to deter them from participating further in the project. Among these respondents, the answer to survey question Q16 showed that patch submission was more difficult than patch creation. While this barrier represents a small fraction of the respondents, it does provide some potential guidance for FLOSS projects who wish to attract more participants. If projects could ease the patch submission process, then they could remove one of the barriers that keeps contributors from continuing.

3) Entry Difficulties: These barriers barriers mostly concerned difficulty with understanding the project. Specific concerns about the project that discouraged further participation included: Unfamiliar project management schemes, lengthy "Help" pages, and complex, poorly documented code. Some respondents expressed frustration with the process of getting themselves ready to contribute. As one noted, "*[i]n general, it took a long time to get a current build running, then the effort from getting the change accepted was more.*"

Other respondents faced difficulties locating the bug they chose to fix within the codebase. These respondents focused mostly on the complexity of the code surrounding the bug and the difficulties in understanding the codebase in general, including the documentation. As one commented, "*extremely time consuming because... require (sic) good understanding of very large codebase.*" Easing the entry process and annotating or better documenting code would lower the barriers to contributing for these respondents.

4) Lack Knowledge: Many of the respondents expressing this view indicated that they were not familiar with the programming language, the bug was above their skill level, or they were not a programmer knowledgeable enough about the code to contribute again. Similar to a concern expressed by those with **entry difficulties**, a few respondents indicated that the code was difficult to understand or that they did not have the specific knowledge to fix the bugs. In general, these respondents simply did not have the knowledge or experience to write code at the level required by the project. As one respondent commented, "*mainly I am not a programmer... anything more complex and my coding skills wouldnt have been up for the job.*" These respondents contributed what they could, and could not contribute any more.

D. RQ4: If no barriers stand in the way of One-Time Contributors, why do they choose to leave after a single patch?

While it is expected that some OTCs have no desire to become long-term contributors, their reasons can still be of interest. Figure 7 shows the distribution of responses given by those whose departure from the project was not the result of any specific barrier (Q17). "Nothing else to contribute" means the respondent had nothing else to add to the project, but would contribute again if they identified something in the future. "Employer" means the respondent contributed as part of their work, but the employer did not need them to make

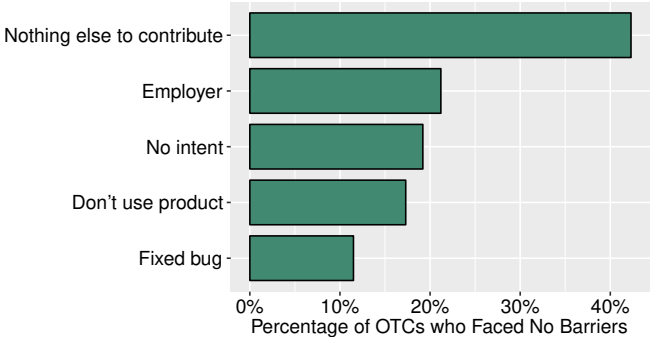

Fig. 7. Non-Barriers Preventing OTCs From Further Code Contributions

additional contributions. "No intent" represents respondents who specifically said they never had any intent to become a contributor. "Don't use product" covers respondents who stopped contributing because they no longer use the FLOSS project. Finally, "fixed bug" represent those who only contributed to fix a specific bug they identified.

Overall, these respondents fixed the problem they saw and moved on to other things. As one of the respondents put it, "*[t]he main reason I haven't contributed more is that it already has what I need and I don't see anything else that needs to be added for my use case*." The project had a bug that was impacting the respondent. The respondent fixed the bug. Having fixed it, the respondent did not have any desire or motivation to contribute anything else to this project.

VI. Discussion

This section discusses the results presented in the previous section and compares those results with results from previous studies to distinguish OTCs from other peripheral developers.

A. Impressions

Prior studies suggest that peripheral developers may be dissuaded from a project if they form negative impressions of project members [3], [24]. However, we found that most of the OTCs were not scared off by negative impressions. Instead, they largely hold positive impressions of the project members. Only a few of the respondents had negative impressions due to the lack of responsiveness. But these respondents acknowledged that the projects were underresourced and the members probably did not have time to respond to outsiders.

B. Motivations

Aside from reciprocity, all motivations discovered in this study were need-based or extrinsic motivations [18], focused on how the project could benefit the respondent. Reciprocity, the only common intrinsic motivation, is not a fun/enjoyment-based intrinsic motivation usually found in hobbyists. Rather, it is an obligation/community-based intrinsic motivation. Similar to other peripheral developers, OTCs were motivated most commonly by extrinsic motivations or community-oriented motivations [21]. In Pinto et. al's study of the casual contributors, the top ranked motivation was "scratching an itch" (i.e., fixing issues that were blocking a developer) [20]. Although, OTCs are a subset of the casual contributors, we also found "fixing a bug" as the top motivations for the OTCs. OTCs' motivations are more similar to need-based contributors than to hobbyist contributors; they contribute primarily for their own needs or out of a sense of obligation to give back to the community. The demographics of the OTCs provide an explanation for this result. In comparison with previous studies examining the demographics of the FLOSS developers [12], [15], OTCs are generally more experienced software developers and have made more contributions to other FLOSS projects.

In comparison with previous studies [3], [5], [18], our sample has fewer paid contributors. Paid contributors have little motivation to continue to contribute to a project unless their employer requires it. Several of the paid OTCs indicated that they would have continued if their job had required it. Almost half of those who reported being paid to contribute listed their employer as their primary motivation.

About a quarter of the respondents reported their employer being their primary motivation, and yet did not reply 'yes' when asked if they were being paid to contribute in Q12. For some, this is because they were working freelance. For others, they might have neglected to reply 'yes' because they contributed *because of* work, and not *for* work. As an example, if the software they used for work was broken or was missing a key feature, they might have fixed the bug or added the feature not for their employer, but because they had to use it for work. These respondents may not have perceived this work as being 'paid for,' as their employer did not directly ask them to contribute to a FLOSS project. Thus, their motivation was both their employer and the desire to fix a bug. The results support this observation. Fixing a bug was the second most common motivation among these contributors, at 30%.

There is little FLOSS projects can do to retain OTCs who contributed their code fully aware that they did not plan to become long-term participants. However, approximately 20% of the OTCs contributed code out of curiosity, for fun, for personal reputation, to solve puzzles, or to experience open source development. Yet they did not make a subsequent code contribution. These OTCs could be considered to be the ones mostly likely to be attracted to the project.

Examining these OTCs in more detail showed that 25% reported negative opinions of the project as a result of being neglected or the unresponsiveness of project members. This findings coincide with those of a prior study's findings that peripheral developers may have to wait 2-19 times more than the core members to get their patches accepted [4]. In addition, 27% of these OTCs indicated that they faced barriers related to the bureaucratic process, licensing terms, and time constraints.

C. Barriers

While previous studies found evidence that newcomers were discouraged from joining a project due to being ignored [24], [31], our results showed a different pattern. Being ignored was one of the less common barriers reported by the OTCs, accounting for only 7% of responses. It is possible that many

OTCs never intended to become long-term contributors, so they had no problem with being ignored. Another possibility is that the projects we studied had a more complicated submission process, making the process a more important barrier.

Additionally, while previous studies showed that the process of identifying a bug to fix and writing or locating test files were difficult barriers faced by FLOSS newcomers [24], [29], none of the OTCs mentioned either as a barrier. As reported in Section V-B, most OTCs were motivated by the desire to fix a bug they personally faced in using the product and only began attempting to contribute after they found a bug. Rather than needing assistance to identify a bug, identifying one was the motivation to contribute in the first place. Instead of difficulties in identifying a bug or in writing tests, our respondents found more difficulties in the complexity of the patch submission process and in the slow turnaround time on code reviews.

Some OTCs did not report any specific problems. They simply had more motivation to leave than stay with the project. However, some contributors did stop contributing due to barriers. While the most common barrier was time (similar to Pinto et. al's study of casual contributors [20]), several other barriers stood in their way, some of which the projects can address. The patch submission process and difficulties with entering the project are both such barriers. It might be expected that newcomers would have difficulties with these steps. However, the survey respondents were relatively experienced in FLOSS projects. Since the respondents were familiar with FLOSS development, it is interesting that they found the process of dealing with the project difficult. Projects that wish to gain more OTCs should try to ease their patch-submission process and entry process, especially since even experienced FLOSS developers had trouble with these processes.

VII. Threats to Validity

This section describes the internal, external, and construct validity threats of the study.

A. Internal Threats

In terms of *selection*, it is possible that some survey recipients were not actually OTCs because they contributed under another alias that we did not identify. A few survey recipients responded that we had incorrectly identified them as an OTC. However, based on the qualitative results and responses to open-ended questions, we do not find this threat to be significant and believe that the vast majority (if not all) of those who responded to the survey were OTCs. Another potential threat is the low response rate. If we calculate out of 3,500, then it is approximately 6%. We cannot be sure that all 3,500 emails actually reached a recipient; even so, the 184 responses we received provide a rich source of data to reveal the insights described in this paper.

B. External Threats

The survey sample may not be representative of all OTCs across all FLOSS projects. The projects used in this study were primarily large, mature projects. The characteristics of OTCs in smaller projects may be different from those included in this study. However, many previous papers have focused on only large, mature projects [18], [21], [22], [30], [31] which gives this choice of using only large, mature projects some validity. Additionally, our results could suffer from a potential bias related to self-selection. The respondents who chose to participate may not be an accurate representation of all OTCs.

C. Construct Threats

It is always possible that survey respondents misunderstand the survey questions. To mitigate this threat, we conducted a pilot study with experts in FLOSS and survey design. We updated the survey based on the results. Additionally, the actual responses to the open-ended questions indicate that the respondents understood the intent of the questions.

VIII. Conclusion

In this paper we defined a type of FLOSS developer that has not been studied, the OTC. We carefully identified 4,127 OTCs and surveyed them to understand their impressions, motivations, and barriers with regards to code contributions in FLOSS projects. The results show that OTCs are indeed distinct from the larger set of peripheral developers. OTCs have high software development experience. Most OTCs are also experienced in contributing to other FLOSS projects and are familiar with FLOSS development.

Interestingly, most OTCs did not have the prior motivation to become long-term contributors. Most of them simply fixed bugs that were impeding their work and wanted to share that fix with the community. Due to their experience and domain knowledge, they faced different barriers from those prior research identified as barriers newcomers faced. However, similar to other newcomers, many OTCs also formed negative impressions of the project due to lack of responsiveness. Yet many of them also understood that FLOSS project members were overworked and those negative impressions were not the primary reason for leaving the project.

However, some OTCs did leave the project because of negative impressions and barriers that FLOSS projects could address to improve the chances of retaining these OTCs. For example, a delayed response to a patch submission bothers many newcomers and OTCs. FLOSS projects should should have means to ensure timely feedback for newcomers. Finally, the patch submission process was difficult even for experienced developers like the OTCs. Therefore, FLOSS projects should provide better documentation to guide newcomers through the patch submission process.

Acknowledgment

This research is partially supported by the US National Science Foundation Grant No. 1322276. Any opinions expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. We thank the respondents to our survey.

References

[1] M. Y. Allaho and W. C. Lee, "Analyzing the social ties and structure of contributors in open source software community," in *Advances in Social Networks Analysis and Mining (ASONAM), 2013 IEEE/ACM International Conference on*, Aug 2013, pp. 56–60.

[2] C. Bird, A. Gourley, P. Devanbu, M. Gertz, and A. Swaminathan, "Mining email social networks," in *Proceedings of the 2006 International Workshop on Mining Software Repositories*, ser. MSR '06, 2006, pp. 137–143. [Online]. Available: http://doi.acm.org/10.1145/1137983.1138016

[3] A. Bosu, J. Carver, R. Guadagno, B. Bassett, D. McCallum, and L. Hochstein, "Peer impressions in open source organizations: A survey," *Journal of Systems and Software*, vol. 94, pp. 4 – 15, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0164121214000818

[4] A. Bosu and J. C. Carver, "Impact of developer reputation on code review outcomes in oss projects: An empirical investigation," in *Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, ser. ESEM '14, 2014, pp. 33:1–33:10.

[5] A. Bosu, J. C. Carver, C. Bird, J. Orbeck, and C. Chockley, "Process Aspects and Social Dynamics of Contemporary Code Review: Insights from Open Source Development as well as Industrial Practice at Microsoft," *IEEE Transactions on Software Engineering (TSE)*, 2016.

[6] A. Bosu, J. C. Carver, M. Hafiz, P. Hilley, and D. Janni, "Identifying the characteristics of vulnerable code changes: An empirical study," in *Proceedings of the 22Nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, ser. FSE 2014. New York, NY, USA: ACM, 2014, pp. 257–268. [Online]. Available: http://doi.acm.org/10.1145/2635868.2635880

[7] J. Cohen, "A coefficient of agreement for nominal scales," *Educational and Psychological Measurement*, vol. 20, no. 1, pp. 37–46, 1960.

[8] K. Crowston, H. Annabi, and J. Howison, "Defining open source software project success," *Proceedings of the 24th International Conference on Information Systems*, pp. 327–340, 2003.

[9] T. T. Dinh-Trong and J. M. Bieman, "The freebsd project: A replication case study of open source development," *IEEE Transactions on Software Engineering*, vol. 31, no. 6, pp. 481–494, 2005.

[10] N. Ducheneaut, "Socialization in an open source software community: A socio-technical analysis," *Comput. Supported Coop. Work*, vol. 14, no. 4, pp. 323–368, Aug. 2005. [Online]. Available: http://dx.doi.org/10.1007/s10606-005-9000-1

[11] J. L. Fleiss, B. Levin, and M. C. Paik, *The Measurement of Interrater Agreement*. John Wiley & Sons, Inc., 2004, pp. 598–626. [Online]. Available: http://dx.doi.org/10.1002/0471445428.ch18

[12] R. A. Ghosh, R. Glott, B. Krieger, and G. Robles, "Free/libre and open source software: Survey and study," 2002.

[13] B. Glaser and J. Strauss, *The Discovery of Grounded Theory: Strategies for Qualitative Research*. Aldine, 1967.

[14] C. Hannebauer, M. Book, and V. Gruhn, "An exploratory study of contribution barriers experienced by newcomers to open source software projects," in *Proceedings of the 1st International Workshop on CrowdSourcing in Software Engineering*, ser. CSI-SE 2014, 2014, pp. 11–14.

[15] A. Hars and S. Ou, "Working for free? motivations of participating in open source projects," in *Proceedings of the 34th Annual Hawaii International Conference on System Sciences*, 2001, pp. 9–pp.

[16] G. G. K. J. Richard Landis, "The measurement of observer agreement for categorical data," *Biometrics*, vol. 33, no. 1, pp. 159–174, 1977. [Online]. Available: http://www.jstor.org/stable/2529310

[17] C. Jensen, S. King, and V. Kuechler, "Joining free/open source software communities: An analysis of newbies' first interactions on project mailing lists," *Proceedings of the 47th Hawaii International Conference on System Sciences*, vol. 0, pp. 1–10, 2011.

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

[19] A. Mockus, R. T. Fielding, and J. D. Herbsleb, "Two case studies of open source software development: Apache and mozilla," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 11, no. 3, pp. 309–346, 2002.

[20] G. Pinto, I. Steinmacher, and M. A. Gerosa, "More common than you think: An in-depth study of casual contributors," in *2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, vol. 1, March 2016, pp. 112–123.

[21] P. Setia, B. Rajagopalan, V. Sambamurthy, and R. Calantone, "How peripheral developers contribute to open-source software development," *Info. Sys. Research*, vol. 23, no. 1, pp. 144–163, Mar. 2012. [Online]. Available: http://dx.doi.org/10.1287/isre.1100.0311

[22] S. K. Shah, "Motivation, governance, and the viability of hybrid forms in open source software development," *Manage. Sci.*, vol. 52, no. 7, pp. 1000–1014, Jul. 2006. [Online]. Available: http://dx.doi.org/10.1287/mnsc.1060.0553

[23] B. Shibuya and T. Tamai, "Understanding the process of participating in open source communities," in *Proceedings of the 2009 ICSE Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development*. IEEE Computer Society, 2009, pp. 1–6.

[24] I. Steinmacher, T. Conte, M. A. Gerosa, and D. Redmiles, "Social barriers faced by newcomers placing their first contribution in open source software projects," in *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, ser. CSCW '15, 2015, pp. 1379–1392. [Online]. Available: http://doi.acm.org/10.1145/2675133.2675215

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

[26] G. von Krogh, S. Spaeth, and K. R. Lakhani, "Community, joining, and specialization in open source software innovation: a case study," *Research Policy*, vol. 32, no. 7, pp. 1217 – 1241, 2003, open Source Software Development. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0048733303000507

[27] J. Wang and A. Sarma, "Which bug should i fix: Helping new developers onboard a new project," in *Proc. CHASE*. ACM, 2011, pp. 24–27.

[28] J. Wang, P. C. Shih, Y. Wu, and J. M. Carroll, "Comparative case studies of open source software peer review practices," *Information and Software Technology*, vol. 67, pp. 1 – 12, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0950584915001068

[29] V. Wolff-Marting, C. Hannebauer, and V. Gruhn, "Patterns for tearing down contribution barriers to floss projects," in *Intelligent Software Methodologies, Tools and Techniques (SoMeT), 2013 IEEE 12th International Conference on*, Sept 2013, pp. 9–14.

[30] Y. Ye and K. Kishida, "Toward an understanding of the motivation of open source software developers," in *Proceedings of the 25th International Conference on Software Engineering*. IEEE, 2003, pp. 419–429.

[31] M. Zhou and A. Mockus, "Who will stay in the floss community? modeling participant's initial behavior," *IEEE Transactions on Software Engineering*, vol. 41, no. 1, pp. 82–99, Jan 2015.