Who Will Stay in the FLOSS Community? Modeling Participant's Initial Behavior

Minghui Zhou and Audris Mockus, *Member, IEEE*

Abstract—Motivation: To survive and succeed, FLOSS projects need contributors able to accomplish critical project tasks. However, such tasks require extensive project experience of long term contributors (LTCs). **Aim:** We measure, understand, and predict how the newcomers' involvement and environment in the issue tracking system (ITS) affect their odds of becoming an LTC. **Method:** ITS data of Mozilla and Gnome, literature, interviews, and online documents were used to design measures of involvement and environment. A logistic regression model was used to explain and predict contributor's odds of becoming an LTC. We also reproduced the results on new data provided by Mozilla. **Results:** We constructed nine measures of involvement and environment based on events recorded in an ITS. Macro-climate is the overall project environment while micro-climate is person-specific and varies among the participants. Newcomers who are able to get at least one issue reported in the first month to be fixed, doubled their odds of becoming an LTC. The macro-climate with high project popularity and the micro-climate with low attention from peers reduced the odds. The precision of LTC prediction was 38 times higher than for a random predictor. We were able to reproduce the results with new Mozilla data without losing the significance or predictive power of the previously published model. We encountered unexpected changes in some attributes and suggest ways to make analysis of ITS data more reproducible. **Conclusions:** The findings suggest the importance of initial behaviors and experiences of new participants and outline empirically-based approaches to help the communities with the recruitment of contributors for long-term participation and to help the participants contribute more effectively. To facilitate the reproduction of the study and of the proposed measures in other contexts, we provide the data we retrieved and the scripts we wrote at https://www.passion-lab.org/projects/developerfluency.html.

Index Terms—Long term contributor, open source software, issue tracking system, mining software repository, extent of involvement, interaction with environment, initial behavior

1 Introduction

To function, Free-Libre and/or open source software (FLOSS) projects need ongoing contributions from new and existing participants [1], [2]. Contributions often involve tasks unrelated to writing code. For example, Lakhani and von Hippel investigated help-provisioning on Usenet in Apache [3]. Similar to Usenet, an issue tracking system (ITS) is necessary to collect user feedback in FLOSS (and other) projects. Unlike Usenet, ITS is used for most project tasks, e.g., reporting, triaging, and fixing bugs (issues) [4], [5], [6], capturing and tracing requirements and modifications to code [7], [8]. ITS contributors participate in these tasks to help resolve issues [9], [10], [11]. Despite being more than an order of magnitude more populous,[1] the ITS contributors have been rarely studied.

To fill this gap and to understand this large group of participants we study contributors to issue resolution. We define a contributor to be any participant in the ITS, not just the source code contributor—a group that has been extensively studied in the past. A contribution is any task involving the use of the ITS.[2]

The start of participation in a FLOSS project is fraught with difficulties [2], [12], as the new contributors may not be familiar with project's practices and norms and the existing participants[3] have to rely on the scant information in a bug report or a comment made by the newcomer to judge the competence and reliability of the new contributor. In contrast, the experienced contributors have become familiar with project practices and norms and have worked (and established rapport) with other participants. Developers with multi-year participation in a project have been found to accomplish more and more important tasks, to provide greater value to the community than others, and are critical to the long-term viability of the community [13], [14]. We believe that by contributing for a long time to ITS, non-developer participants also gain experience and become more valuable.

We, therefore, want to understand what affects the chances that a new contributor would stay with the project and what the existing project community or new

1. For example, for Mozilla, 6 K individuals committed to Version Control System (http://hg.mozilla.org/, retrieved January 2014), while 210 K individuals participated in ITS (http://people.mozilla.org/mhoye/bugzilla/, retrieved January 2013).

• *M. Zhou is with the School of Electronics Engineering and Computer Science, Peking University and Key Laboratory of High Confidence Software Technologies, Ministry of Education, Beijing 100871, China. E-mail: zhmh@pku.edu.cn.*
• *A. Mockus is with the Department of Electrical Engineering and Computer Science, University of Tennessee, Min H. Kao Building, Room 613, 1520 Middle Drive, Knoxville, TN 37996-2250. E-mail: audris@utk.edu.*

Manuscript received 17 Dec. 2013; revised 30 July 2014; accepted 10 Aug. 2014. Date of publication 18 Aug. 2014; date of current version 14 Jan. 2015. Recommended for acceptance by R. Jeffery.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TSE.2014.2349496

2. Note that ITS participants may also contribute code to the code repository.
3. In this paper we use contributor and participant interchangeably.

contributors could do to achieve that. Specifically, we would like to address a practical question: can the initial activities reveal who would stay with a project for a long time?

We refer to such committed long term ITS contributors as LTCs. More specifically, we define LTC to be a contributor who stays with the project for at least three years (from their first activity recorded in the ITS) and who has productivity (measured via issues modified per year) exceeding 10th percentile among the participants with a tenure exceeding three years. We added the second condition to exclude contributors who may stay with the project for a long time but contribute little.

The ability to predict LTCs and the measures that affect the odds of becoming an LTC may help the community to focus its limited resources. For example, the number of new participants may exceed the community's newcomer mentoring capacity [10]. By revealing the most likely future LTCs, the prediction would help focus mentoring on these promising candidates. The relationships revealed by the prediction model may also help determine ways to improve the initial experiences of new contributors and may reveal an explicit path to train and retain new participants until they become capable of solving complicated tasks effectively and accurately. We use the ITS data to model the initial behavior of a new participant: we construct the predictors of long term participation and model the relationship between the predictors and the probability of becoming an LTC.

The rest of the paper is organized as follows. We visit related work in Section 2 and describe the data-driven process employed by this study in Section 3. We borrow the existing theories to construct predictors of long term participation derived from ITS data in Section 4. Based on the predictors, we model and predict the probability of a new participant becoming an LTC, and reproduce the model in Section 5. We conduct a survey to validate the data and the results in Section 6. We discuss the limitations in Section 7, and conclude in Section 8.

2 Related Work

The existing literature provides a number of theories and measures that may lead individuals to engage in FLOSS projects, e.g., [10], [15], [16], [17], [18], [19], [20], [21], [22], [23]. We could classify the relevant predictors into two groups: motivation-related concepts and environmental variables, with more details provided in Section 4.1. Most of these measures are derived from interviews, surveys, online project documents, and online discussion groups. A few of the concepts were quantified through archival software development data (e.g., version control data, as in [19], [24], [25]) and even fewer via ITS data. The most relevant measures derived from the ITS were the number of completed issues measuring team effort [22] and productivity [24], the number of defects fixed measuring individual performance [25], and the relative sociality (RS) [10]. Furthermore, even for these ITS-derived measures non-developer contributors have been rarely studied despite the fact that non-developer contributors do accomplish many tasks in FLOSS projects [6].

TABLE 1
Projects

Project	Years	MLOC[1]	Domain	Cntrbtrs
Gnome		7.9	UI	156,332
Evolution	10	0.8	Calendar and Mailbox	21,041
Nautilus		0.1	File manager	17,430
Epiphany		0.1	Browser	3,716
Mozilla		20.0	UI	187,333
Firefox	12	5.3	Browser	47,690
Thunderbird		1.1	Mailbox	12,993
Calendar		0.8	Calendar	4,130

[1] *Data from ohloh.net.*

A number of publications tackle a related phenomena of sustainability, e.g., how a sustainable group evolves [26], how online community should encourage commitment [27], how the successful OSS participants progressively enroll a network of human and material allies to support their efforts [28], and how the congruence of values between the individual and their organization affects turnover [29]. Unlike in prior work, we model how the initial behavior and experiences of a new contributor are related to the chances that they would stay for a long term with a project.

Though our research focus and method differ from the existing literature, we use some of the literature to structure and interpret the predictors of long-term participation created from the ITS data as described in Section 4.

3 Methodology

The ITS records the history of how people initiate and complete various tasks in software projects. Such detailed data may contain traces of behavior that indicate if a new participant will continue to contribute for a long time. We, therefore, use ITS data to investigate participants' initial behavior in two FLOSS projects—Mozilla and Gnome. While we perform a more in-depth analysis on Mozilla, we use Gnome to ensure that the findings on Mozilla are applicable more generally.

We start from introducing the projects and their issue workflow in Section 3.1, and present our data-driven analysis method in Section 3.2.

3.1 Context

3.1.1 Mozilla and Gnome

Mozilla and Gnome implement user interface functionality, and have more than 10 years of history, as shown in Table 1. Both contain a number of sub-projects, with major sub-projects presented in the table. Evolution is the largest Gnome project, and Firefox is the largest Mozilla project. Note that both ecosystems implement basic desktop tools such as a browser and an email client.

3.1.2 Issue Workflow

In FLOSS projects the issue tracking systems not only track tasks for developers and testers, but also track issues raised by end users and by the down-stream projects, e.g., Ubuntu. Both Mozilla and Gnome use Bugzilla ITS to track issues.

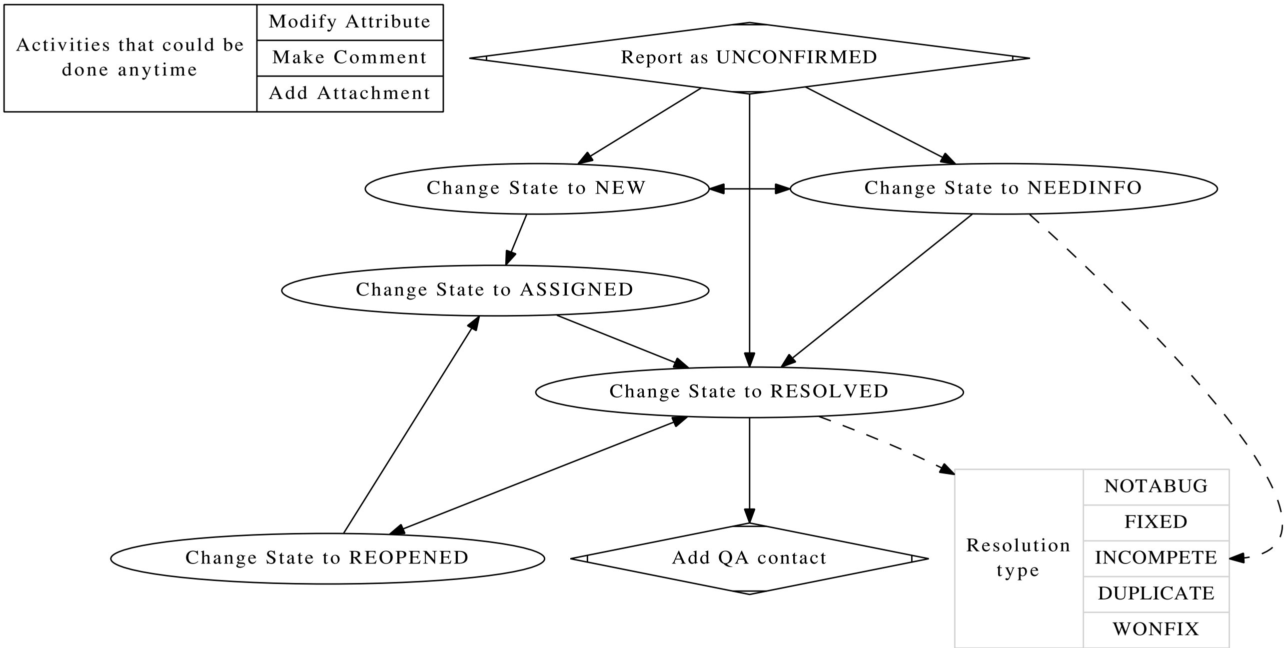

Fig. 1. Typical issue workflow.

FLOSS projects often define a protocol to follow when tracking issues. In particular, Gnome defines the standard steps on its website[4] (Mozilla's issue process is similar[5]). According to it, an issue (for brevity we interchangeably refer to issue as Modification Request (MR), a term borrowed from industry) is reported (born) in an UNCONFIRMED state. The issue may be confirmed by setting its state to NEW or, it may be immediately resolved with the state set to RESOLVED. To indicate that additional information from the issue reporter is needed to proceed further in fixing this issue, the state is set to NEEDINFO. When the issue is assigned to a developer, the state is set to ASSIGNED. Each "RESOLVED" issue has a resolution, e.g., FIXED, DUPLICATE, INCOMPLETE. As the name suggests, FIXED means the bug is fixed, DUPLICATE means the reported bug is a duplicate of some other bug, and INCOMPLETE means the reported information is not sufficient to reproduce the bug.

Apart from the state changes, there are other activities recorded by the ITS. For example, a participant might comment on an MR for any reason at any time, modify an attribute, such as Severity or Operating System, and add an attachment of a debugging trace or a patch. At some point in MR's lifetime a quality assurance contact (QA contact) is often added to address potential later concerns.

Fig. 1 summarizes the typical issue workflow. From the time somebody reported an MR until the time somebody closed it (it also may remain open at the time of the study), a sequence of events take place: MR is created, assigned, submitted, tested, and resolved. It may also be reassigned, its attributes changed, comments, debugging traces, QA contact etc. added. Each such event has an associate date, time, the type of action, and the contributor performing the action. The transition from one action to another may involve a change of actors. We define these hand-offs to be edges in the project workflow graphs.

4. http://live.gnome.org/Bugsquad/TriageGuide
5. https://wiki.mozilla.org/QA/Triage

3.2 Data-Driven Analysis

We follow the data-driven process [30] to conduct our analysis. The primary step is to obtain, clean, and validate ITS data. A variety of techniques are applied to ensure completeness and accuracy of the various attributes and measures, including the ITS data itself (issue reports) and documents describing practices of ITS use in the projects. The process also involves interviewing contributors and conducting surveys. The interviews are used to clarify unresolved questions and to confirm the findings from the ITS data. The survey is primarily used to confirm that the data cleaning and correction were successful.

More specifically, we start from discovering and retrieving ITS data in Section 3.2.1. After cleaning and processing the raw data (see Section 7.1 and data from earlier studies [6], [10], [31]), we verify that the data are complete. We discuss how to identify participants, the most critical task related to data completeness in this study, in Section 3.2.2. We then use documents, literature, small interviews and surveys to interpret and validate the facts obtained from the data, including understanding ITS practices and participants' initial experiences, as described in Section 3.2.3.

Based on the above exploration, we target our research questions to construct metrics derived from the validated ITS data, and build relationships among the metrics, as described in Sections 4 and 5. Finally, the retrieval of new data (not available at the original date of model construction) provides a realistic test of how the model predictions would work in practice (Section 5.2). A survey sent to 240 contributors is used to interpret the metrics and the model in Section 6. This chain of evidence is illustrated in the chart presented in Appendix C, which can be found on the Computer Society Digital Library at http://doi.

TABLE 2
Six Bugzilla Extracts

Name	Author	URL
Mozilla 2013	Mozilla Community	people.mozilla.org/m̃hoye/Bugzilla
Mozilla 2012	authors	passion-lab.org/projects/developerfluency.html
Mozilla 2011	authors	same
Gnome 2011	authors	same
Gnome 2008	C. Bird	msr.uwaterloo.ca/msr2009/challenge/msrchallengedata.html
Gnome 2006	P. Wagstrom	academic.patrick.wagstrom.net/research/gnome

ieeecomputersociety.org/10.1109/TSE.2014.2349496, and is used to argue that the relationships obtained from the model are based on facts and interpretable measures that predict contributor's tenure with the project.

3.2.1 Retrieving Issue Tracking Data

Bugzilla is used by both Mozilla and Gnome and the issues are accessible on the web. We refer to all data stored in the Bugzilla of Mozilla or Gnome at a particular time as a Bugzilla extract. Because of the higher quality of data in Mozilla's Bugzilla, we conduct the primary modeling and prediction on Mozilla data, and use Gnome as a comparison.[6]

We used six Bugzilla extracts obtained by different individuals at different times shown in Table 2. Three of the extracts were retrieved by the authors. We obtained information for all issues in XML format[7] as well as the activity history[8] for each project in January, 2011. There are not many issues prior to 1998 in Mozilla and very few prior to 1999 in Gnome, hence we removed data before 1998 in Mozilla and before 1999 in Gnome. Overall 200,655 user ids and 620,511 MRs were in Mozilla, and 158,244 user ids and 517,801 MRs were in Gnome. We used these two extracts to do modeling as reported in Section 5.1.

To conduct a realistic validation of the prediction, we retrieved Mozilla Bugzila again in May 2012. This later extract had 214,576 user ids and 709,386 MRs. We used this extract to validate model predictions made earlier as reported in Section 5.3.1. In 2013, Mozilla's Bugzilla dump was offered by the Mozilla community, and provided us with a different type of data to reproduce the model, as reported in Section 5.3.2. Gnome 2008 and Gnome 2006 extracts were used to investigate the quality of data in Gnome 2011 extract.

3.2.2 Identifying Contributors in Bugzilla

Identifying contributors is extremely important in this study. If we couldn't identify participants appropriately, the very foundation of this study would be unstable. Bugzilla identifies participants via email, name, or login, but the record is not perfect and it may change over time. This may lead to the same identification being associated with multiple individuals (multi-person ID) or to several distinct identifications for the same individual (multi-ID person). Therefore we used multiple extracts shown in Table 2 for cross comparison to evaluate different approaches (see Section 7.2). We settled on using an email as a personal ID in Mozilla, and a login as a personal ID in Gnome.[6]

3.2.3 Qualitative Investigation

We start from data exploration, search for and read digital records, and communicate with experts to clarify and validate intermediate findings. That helps us to obtain more accurate results by combining and triangulating information from disparate sources. In particular, we went through the following procedures:

- We read the existing literature, particularly on Mozilla and Gnome, e.g., [1], [23], [32], [33], to understand the project context and practices;
- We inspected project websites looking for the project-related information, e.g., the standard workflow of resolving issues. We also looked at the subproject webpages, searched for relevant information, e.g., the practices used to report and resolve an issue;
- We randomly sampled 40 people (20 non-LTCs and 20 LTCs) from each project and read the issues they were involved in during their first month to understand their contribution practices and experiences.
- We targeted a small group of experts (eight LTCs sampled from each project) to ask what motivated them to report/comment on the issues in the beginning. We selected the interview subjects who had the most influence on other contributors in the project [14], in order to get the most information from a limited number of interviewees. We obtained one reply from each project (five emails couldn't be delivered), and we followed up with additional questions about the activities of new contributors.

As a final interpretation and validation step we conducted a survey on a stratified sample[9] of LTCs and non-LTCs (40 LTCs and 80 non-LTCs from each project) as described in Section 6.

6. It is not possible to retrieve all (or a large number) of issues from Gnome Bugzilla with full email of contributors. Therefore Gnome data to which we have access contain only logins instead of full emails. This makes it many difficulties for some of the analyses we tried to conduct as explained in Section 7.

7. e.g., https://Bugzilla.mozilla.org/show_bug.cgi?ctype=xml&id=3549

8. e.g., https://Bugzilla.mozilla.org/show_activity.cgi?id=3549

9. Stratification is the process of dividing members of the population into homogeneous subgroups before sampling. When sub-populations vary considerably, it is advantageous to sample each sub-population (stratum) independently [34].

4 Constructing Participation Predictors

In this section we aim to answer the following research question:

- *RQ1*. what are the factors that affect the chances for a new contributor to become an LTC?

Enormous effort over the past decades was spent in attempts to understand factors that affect involvement and performance—aspects of behavior that we want to quantify and model. The bulk of that research concerns job performance, not activities done by predominantly volunteer group studied by us. These theories inspired some of the measures of contributor behavior recorded in ITS as described in Sections 4.1 and 4.2.

4.1 Predictors in the Literature

We classify the predictors described in the literature into motivation-related concepts and environmental variables.

The first group involves motivation-related concepts and measures. Blumberg and Pringle [15] recognized capacity, willingness, and opportunity as three interacting dimensions accounting for individual performance. Rasch and Tosi [16] used ability, effort, goal difficulty, goal specificity, and self-esteem as the performance drivers. Hertel et al. [17] used the motivation measures of participation in social movements (collective motives, social motives, reward motives) and the valence, instrumentality, self-efficacy, and trust (VIST) measures of individual motivation in teams to explain the participation in the Linux kernel. Lakhani and Wolf [18] suggested that enjoyment-based intrinsic motivation is the strongest and most pervasive driver, with user need, intellectual stimulation derived from writing code, and improving programming skills being the top motivators for project participation. Roberts et al. [19] found that developers' intrinsic and extrinsic motivations are not independent but rather are related in complex ways. Shah [13] found that a need for software-related improvements drives the initial participation, but only a small subset hobbyists remain involved. Nakakoji et al. [35] found that the willingness to get involved determines the role played by a FLOSS member in the community.

This literature reveals that ability and willingness are critical factors that drive individual engagement. Ability refers to the capabilities that enable an individual to perform a task effectively, including skills, experiences, etc. Willingness refers to the psychological and emotional characteristics that influence the degree to which an individual is inclined to perform a task, including motivation, personality, effort and attitude. However, people's activities represent a combined effect of multiple dimensions. It is, therefore, almost impossible to separate ability from willingness through measurement. Ericsson et al. [36] suggested that an individual's ability and practice are not separable, i.e., the talent is not needed to explain performance if the amount of deliberate practice is taken into account and the only way to increase the amount of deliberate practice is through willingness. We, therefore, encompass the ability and willingness into a single dimension which should determine how much an individual would get involved in a volunteer activity. The extent of involvement is a concern commonly discussed in the FLOSS projects. It is simpler to operationalize, it is a less ambiguous concept, and it represents an explicit expression of participant's willingness and ability. We, therefore, focus on quantifying the participant's extent of involvement in Section 4.2.1.

The second group of predictors involves variables representing the environment. In particular, participant's project climate affects her participation. For example, project's relative sociality was found to be related to participants' retention [10]. Participant's perception about her environment and her interactions with the community impact her outcomes. For example, the perception by the community members of community usefulness is associated with individual activities in virtual communities [20]. The perceived status of the noninitiator members of a project influences its probability of attracting developers [21]. Similarly, the extent to which an individual's values are consistent with those revealed in his or her organization/environment, was found to yield significant effects on a variety of attitudinal outcomes like job satisfaction and organizational commitment, and behavioral outcomes like job performance and turnover [29], [37], [38]. If developers shared the beliefs and norms of the community, they engaged more in the effort related to the community [22], [23]. Similarly, identity-based and bond-based commitment is important for contributor retention [27]. Co-workers (or peers) are particularly important for project/community participants. For example, developers depend on their co-workers to get their frequently sought information [39]. Community participants acquire the skills and knowledge embodied in the community by interacting with master members (by reading their code and by asking them questions) and by practicing authentic yet small tasks [35]. Specific attention from the community such as personalized messages, leads to more powerful effects than generic ones, in which the newcomer receives a standardized message such as a welcome-to-the-project template [27].

This literature highlights the influence of environment on participants, inspiring us to measure the interactions between participant and environment derived from ITS data to predict long tenure in Section 4.2.2. The relationships among concepts, dimensions, and measures are shown in Fig. 2.

4.2 ITS-Derived Predictors

ITS records the activities of contributors, such as reporting and commenting, and may contain traces of behavior that indicate if a new participant will continue to contribute for a long time. We, therefore, derive measures from ITS that have the potential to discriminate between the LTCs and non-LTCs. As noted above, we separate our measures into two dimensions: the extent to which the individuals get involved in helping the project and the interaction between the individuals and their environment, as shown in Table 3.

4.2.1 The Extent of Involvement

We measure the extent of participation in three ways. First we consider *the number of tasks*, e.g., the number of comments a participant makes. These comments might help interpret a confusing report, suggest a possible solution, or explain the benefits of a proposed new feature.

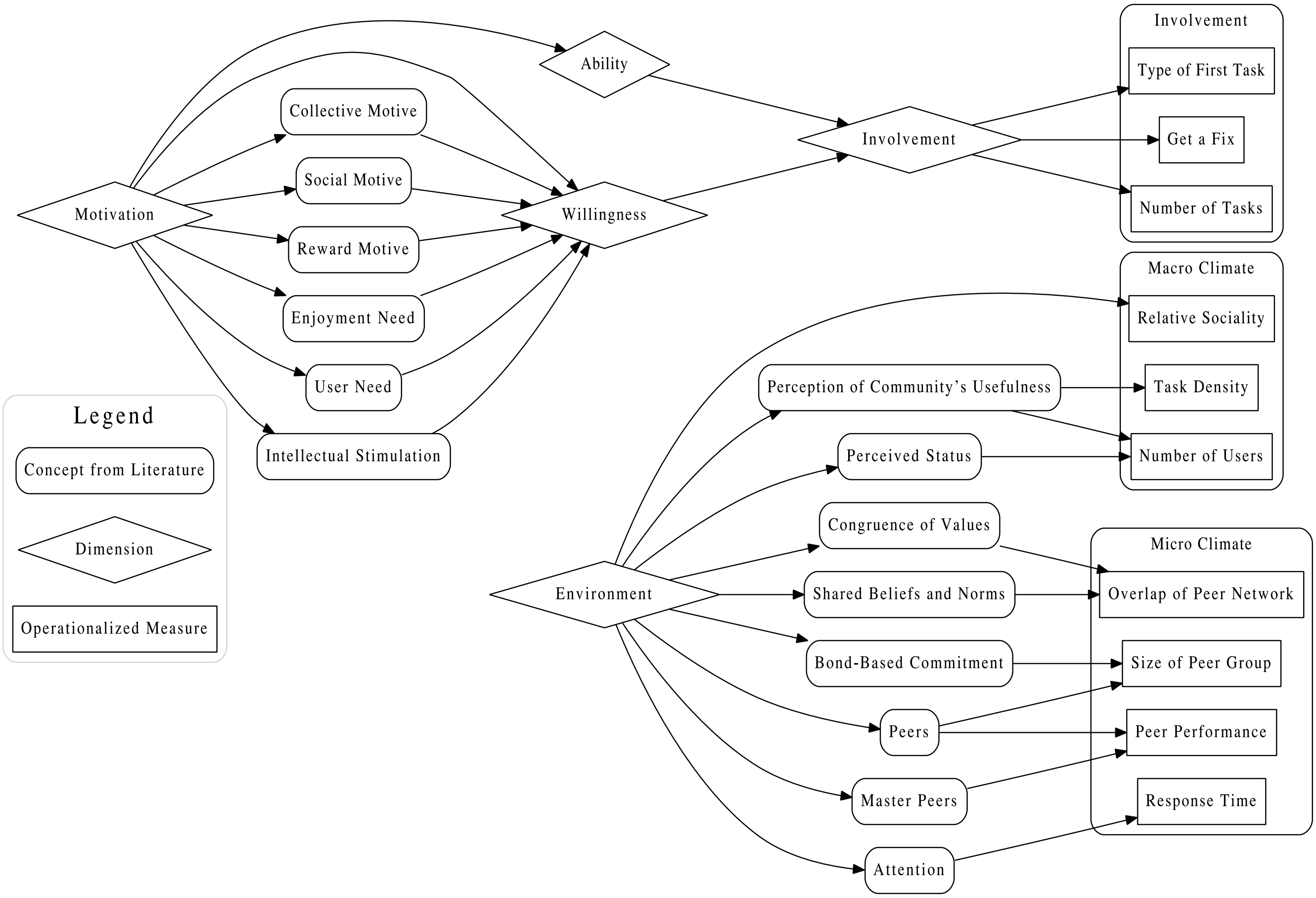

Fig. 2. Relationship between theories and ITS-derived metrics.

The second measure gauges the effort a participant spends on tasks. The measure is an indicator of whether or not the participant gets her first reported issue to be fixed. The more time and effort a contributor spends searching in Bugzilla for issues similar to the encountered issue, the more likely she is to find exactly the same or a similar issue. It is less likely, therefore, that the issue will end up as DUPLICATE (i.e., not fixed). The same argument applies for issues with resolution INCOMPLETE—timely responses with requested information necessary for developers to reproduce the issue, would make it more likely that the issue will be fixed. In other words, if a person provides sufficient effort, the issues she reports have more chances to be fixed and, thus, improve the quality of the product. Consequently, whether or not *the first reported issue gets fixed* is likely to indicate the effort provided by the contributor.

The third measure indicates the differences in the effort it takes to start participation (a barrier to entry) by considering the types of tasks. In particular, *starting participation from a comment* for an existing issue instead of reporting a new issue implies that either the new participant is intentionally trying to help, or, alternatively, they are carefully following the instruction on how to report an issue and have searched for and have discovered an existing issue adding a comment to it instead of simply reporting the problem as a new issue. In contrast, reporting an issue through a crash-reporting tool such as Bug-Buddy is extremely easy and, therefore, does not prove that a participant spent a lot of effort to get involved. A crash of an application that uses Gnome libraries initiates Bug-Buddy interface inviting the user to submit the stack trace.

TABLE 3
Predictors for Participant i

Dimension	Predictor	Description
Extent of Involvement	nCmt	Logarithm of the number of comments $+1$
	GotFix	At least one of the issues reported by i was fixed
	withBB	First report by i uses a crash reporting tool
	FNotRep	i starts participation with a comment
Macro-climate	nUsr	Number of product users when i joins
	RS	Project's relative sociality
Micro-climate	nPeer	Logarithm of i's peers' group size: $\ln \Vert \cup_{p\in Peers(i)} Peers(p) \Vert + 1$, where $Peers(p)$ is the peer group for p
	pShared	Logarithm of the social clustering of i's peer group $\ln \frac{\sum_{p\in Peers(i)} \Vert Peers(p)\Vert}{\Vert\cup_{p\in Peers(i)} Peers(p)\Vert}$
	PeerPerf	Logarithm of the minimum productivity (issues/month) of the peers $\ln min_{p\in Peers(i)} nmr_p + 1$
	LckAttn	The longest duration between the i's action until the response is less than 1 hour

A user simply needs to click "Submit" button to generate an issue report. For comparison, the regular way of reporting an issue involves applying for an account for Gnome Bugzilla, creating a new issue report, and filling in the template that includes steps needed to reproduce the bug. In Gnome, less than 1 percent of the contributors, who had their first issue reported via Bug-Buddy, eventually became LTCs, while more than 4 percent of the contributors who started with a regular Bugzilla report became LTCs.

4.2.2 Interactions between Contributors and Environment

We identified two types of environment a participant encounters in a project. Macro-climate is the overall project environment that is the same for everybody in the project. We attempt to measure several elements of macro-climate: product's popularity, project's task density, and project's relative sociality. The market value of a no-cost product is embodied in its usage. *The number of users* ($nUsr$), therefore, should be related to the product's market value (reflect the monetization opportunities) and that, in turn, will affect the funding (tools, equipment, materials, supplies, and pay) for the project and the degree of interest people are likely to devote to it. Consequently, it will likely affect the contributors' stay with the project. We chose to measure project's popularity via the size of its user population obtained from various sources available online. In particular, Firefox is the primary product in Mozilla and we use its user base as a proxy for Mozilla project user base. We obtained user population by multiplying the estimate of the market share of Firefox[10] by the estimate of the number of Internet users.[11] We were able to obtain Internet user estimates starting from December, 2000. Similarly, we did the following to approximate the historic numbers of Gnome users. First, we obtained the estimates of the fraction of Linux users.[12] Then, we used surveys of desktop choices for the period between 2003 and 2008.[13] For the period from 2009 to 2011 we approximated Gnome users by the fraction of Ubuntu users.[14] Eventually we multiplied the market share of Ubuntu/Gnome by the estimates of Internet users to approximate Gnome user numbers.

Project task density describes how much work is done in the project, and we measure it by the number of active MRs each month ($numMR$). This aspect of macro-climate indicates whether the project is active and whether the participants have high workloads. *Project sociality* represents project's social climate and we could measure it by the participation density, i.e., the number of participants each month ($newJoiner$), or project's relative sociality. RS is the geometric average over all project's participants of the following ratio: the number of individual's workflow peers over the number of MRs that an individual has participated in (during that month). It measures the number of other individuals one has to encounter when resolving an average task. In a macro-environment with a low RS tasks tend to be resolved by a single individual, while the environment with a high RS indicates that tasks tend to involve interactions among many individuals.

We obtain a participant's workflow peers based on the project workflow graph described in Section 3.1.2. More specifically, for all the MRs, we obtain the chronological sequence of tuples containing: date and time, MR ID, contributor ID, action, and value. We then create a link between every pair of contributors immediately adjacent in this sequence. The assumption is that such transfer of MR ownership indicates a likely communication between these contributors. While the verbal communication may rarely accompany such a hand-off in FLOSS projects, it still represents an artifact-mediated communication as illustrated by, for example, [9], [10]. In summary, we consider that contributor Alice encounters contributor Bob if a hand-off is transferred from Bob to Alice. We do not consider the communication to have occurred in the opposite direction because Bob may be, in some cases, not even aware of Alice's existance.

Micro-climate represents the environment of each individual and it, therefore, varies among the participants. In particular, the unique set of people a participant encounters in her workflow network and the interactions with them is a defining feature of micro-climate. In other words, the actions or performance of her workflow peers and her relationship with them constitute the unique micro-climate of the contributor. We chose the initial size of a participant's workflow group, their productivity, their social clustering, and the attention they provide to her as measures of the micro-climate.

The size of a person's workflow peer group, i.e., the number of peers in her workflow network, is primarily determined by her own actions. The more issues she is involved in, the more likely she will encounter additional peers. But the actual act of contribution is associated with the person's accumulated ability and willingness to contribute. At the same time, her peers constitute her social working conditions [21], [27], [35], [39]. We assume that if a person has more peers, she is more likely to attach to the project and, therefore, she is more likely to become an LTC. On the other hand, *the performance of her peers* is likely to affect her performance, for example, Mockus [40] found that more productive mentors lead to more productive followers.

We consider *the social clustering* to be the amount of replication among the workflow networks of peers. For example, contributor Alice has two peers, Dragon and Tiger, and Dragon meets Lion and Bear, while Tiger meets Lion and Deer as shown in Fig. 3. The sum of Alice's two peers' network sizes is $3 + 3 = 6$, but the size of the joint network is 4 (because Lion and Alice are repeated twice), therefore her social clustering is $\frac{6-4}{4}$ (see Table 3 for the formal definition).

The underlying assumption is that if a person's peers have more in common (share more colleagues), it is more likely that they would have similar project experiences and would share similar values. The new participant, hence, is less likely to get confused by a variety of behaviors and value systems she observes. Furthermore, more clustered peer group is more likely to understand and trust each other, and that, in turn, might create a better environment

10. http://en.wikipedia.org/wiki/Usage_share_of_web_browsers
11. http://www.internetworldstats.com/stats.htm
12. http://www.w3schools.com/browsers/browsers_os.asp
13. http://www.desktoplinux.com/news/NS8454912761.html, http://www.desktoplinux.com/articles/AT2127420238.html
14. http://stats.wikimedia.org/archive/squid_reports/

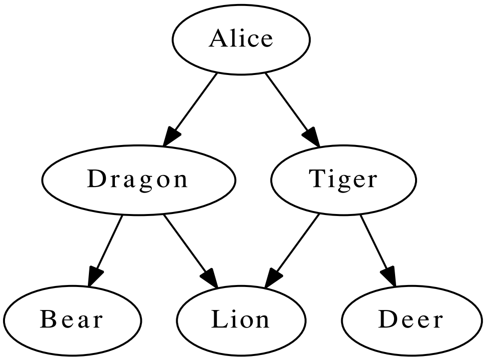

Fig. 3. An example of social clustering.

for a newcomer to learn and to become more effective. It might also increase her work satisfaction and the willingness to stay [22], [23], [27].

Humans need attention from other people, and FLOSS contributors are no exception, notwithstanding common stereotypes. Perhaps, the more attention a newcomer could obtain from the existing project members, the more likely that she would stay with the project. *The duration of time between the newcomer's first action until somebody responds* may reflect the amount of attention. The response delay that is too long or too short may not be perceived as a signal of attention. A long response delay may mean that others either are too busy to respond or feel that the issue is not significant enough to warrant attention. An immediate response may imply that the responder did not take the issue seriously or did not inspect it carefully but just replied with a canned template to save time, e.g., *"Thanks for taking the time to report this bug. This bug report isn't very useful because it doesn't describe the bug well."* In these circumstances the reporter might feel under-appreciated and stop contributing.

5 Modeling and Predicting LTCs

In this section we model if the variation in the extent of involvement and environment at the time contributors join is related to who will (and will not) become an LTC. Specifically, we try to answer the following research question:

- *RQ2.* Does the extent of involvement and environment predict whether a new contributor will stay for a long time?

We also investigate methodological questions related to the reproducibility of ITS-derived results. The properties of nonreproducible phenomena cannot provide a basis for theories in software engineering [41]. We therefore used a recent Mozilla-provided Bugzilla database dump to answer RQ3:

- Can we reproduce the measures constructed in the earlier study based on newly retrieved data?
- Were the predictions published earlier accurate?
- Was there a "publication bias" [42], i.e., did the significance of the coefficients decrease with the additional observations?

We fit a logistic regression model of the probability that a newcomer will become an LTC in Section 5.1, and predict which participants will become LTCs in Section 5.2. We check for "publication bias" in Section 5.3.

5.1 Modeling the Chances of an Individual Becoming an LTC

5.1.1 Operationalizing Model

We investigate the influence of the extent of involvement and environment on the chances of an individual's success in the project by fitting a logistic regression model specified in Equation (1). The response is the indicator of a new participant becoming an LTC (the proxy for success) and the predictors include measures of the extent of her involvement and measures of her macro- and micro-climate as described in Section 4 (Not all factors are included in the model because of the correlations, e.g., project task density is highly correlated with RS).

The model is fitted based on the observations derived from Mozilla 2011 extract and Gnome 2011 extract. Each observation represents one project participant, with the predictors calculated over her first month from joining shown in Table 3. We considered a variety of intervals from one week to six months. Intervals shorter than one month do not provide a good model fit or prediction, while longer intervals did not improve prediction or fit substantially. There are 125,665 observations in Gnome and 130,471 observations in Mozilla.

The predictors that require more explanation are discussed below.

We operationalized the size of peer group ($nPeer$) in two ways. The first approach counts the number of other participants encountered during the first month. Second approach considers the number of participants encountered by her peers. Both measures have a similar association with the response, but we present the second measure because it explains more variance in the response and has lower correlations with other predictors in both projects.

To represent the effort a contributor provides to the community we used $GotFix$, an indicator of having at least one of the reports to be fixed.

Barrier to entry BtE depends on the project. We used $FNotRep$ (the first participation is not an issue report) for Mozilla and $withBB$ (the first participation is using BugBuddy) for Gnome, because $withBB$ explains more deviance than $FNotRep$ in Gnome. Since Mozilla did not have an equivalent tool that required minimal effort to report an issue we used $FNotRep$ as a proxy of the high level of involvement.

$PeerPerf$, is the lowest performance over all peers and measured by the number of MRs modified by the peer during that month.

We used $LckAttn$ to represent an extreme situation of a too-rapid response (within one hour). We have tried a variety of operationalizations of this measure. For example, we also considered a too-slow response (more than 24 hours), but we chose to report the too-rapid response because it has a similar impact in both projects.

Predictor prj is a sub-project indicator of the ecosystem the participant starts with, e.g., Evolution in Gnome, Firefox in Mozilla. It explains 1-2 percent of the total deviance and was added to account for the variation among sub-project

TABLE 4
Model for Mozilla (130,471 Observations)

	Est	Std.Err.	x	x_{alt}	$\frac{Odds(x_{alt})}{Odds(x)}$
(Intcpt)	−7.49	0.419			
nUsr	−0.601	0.15	0.308	0.57	85/100
RS	0.701	0.293	0.0738	0.173	107/100
GotFix	0.74	0.0831	F	T	210/100
FNotRep	0.507	0.0821	F	T	166/100
nCmt	0.819	0.0409	ln 2	ln 5	212/100
nPeer	0.142	0.0205	ln 1650	ln 4266	114/100
pShared	2.35	0.135	ln 1.12	ln 1.32	148/100
LckAttn	−0.325	0.124	F	T	72/100
PeerPerf	0.0649	0.0131	ln 15	ln 249	120/100

TABLE 5
Deviance for Mozilla Model

	Df	Deviance	Dev.Percent(%)	P-value
NULL	130,471	13,408		
nUsr	1	68	0.51	5.77e-05
RS	1	92	0.69	0.0167
GotFix	1	298	2.22	5.40e-19
FNotRep	1	53	0.4	6.34e-10
nCmt	1	1479	11.03	2.39e-89
nPeer	1	120	0.89	4.10e-12
pShared	1	264	1.97	1.82e-67
LckAttn	1	5	0.04	0.00878
PeerPerf	1	19	0.14	7.52e-07
prj	15	195	1.45	

environments. Note that we tried to include a control for time to adjust for the possibility that the conversion rate may vary over time. However, it was highly correlated with the measures of environment ($nUsr$), therefore we could only employ one of the predictors. Project popularity ($nUsr$) increased over time, and, therefore, it can be interpreted both as project popularity or as a proxy of calendar time. The modeling results obtained by replacing $nUsr$ by calendar time are similar.

$$\begin{aligned} isLTC \sim nUsr + RS + GotFix + BtE \\ + nCmt + nPeer + pShared \\ + LckAttn + PeerPerf + prj. \end{aligned} \tag{1}$$

5.1.2 Modeling Results

Tables 5 and 7 show the deviance explained for Mozilla and Gnome,[15] 19 percent of the deviance is explained in Mozilla model and 23 percent in Gnome. Tables 4 and 6 contain fitted values. The second column has the estimated coefficients, and the third standard errors. All predictors are significant (at <0.005 level), except for RS and $LckAttn$ in Mozilla (p-values of 0.02 and 0.009), which are not significant. We follow Johnson's [43] recommendation to use p-value of 0.005 for statistical evidence instead of the commonly used value of 0.05, because using the latter value often leads to unreproducible results. When reproducing Mozilla results with more data (see Section 5.3) we, indeed, have found that the p-value for RS was no longer significant even at a 0.05 level. However, the p-value for $LckAttn$ decreased to $5e-6$. In the discussion below we do not consider that we have evidence for RS to be explaining the response for Mozilla, but we consider that we have evidence (based on replication) that $LckAttn$ is explaining the response for Mozilla.

The fourth to sixth columns show practical importance of the predictor in determining the LTC probability through effect sizes. In logistic regression effect sizes for continuous predictors ($nUsr$, RS, $pShared$, and $PeerPerf$) are represented by odds ratio of the estimate for the predictor (column labeled x) over the estimate plus the standard deviation (column labeled x_{alt}). To measure the effect size for discrete predictors with few distinct values ($nPeer$ and $nCmt$) we chose to use median and 75th or 90th percentiles to make the interpretation of the effect size more meaningful. For the boolean predictors such as $GotFix$, $withBB$, $FNotRep$, and $LckAttn$ the effect size is the odds ratio for the most frequent and the less frequent values. To illustrate effect sizes, let's consider two hypothetical Mozilla contributors: Alice with one comment during her first month and Bob with four comments (90th percentile of $nCmt$ is $\ln(4+1)$ or four comments). The odds for Bob to become an LTC are 112 percent higher than odds for Alice if their remaining predictors have values specified in the forth column of Table 4.

The models show that a contributor's extent of involvement and interaction with environment affect her odds to become an LTC. Specifically, starting from comments instead of reports, reporting via Bugzilla instead of BugBuddy, or reporting an issue that gets fixed double the odds of becoming an LTC. The micro-climate environment with low attention in the form of a too rapid response reduces the odds by 28 percent in Mozilla and by 39 percent in Gnome. Increase of the productivity of the slowest peer from 14 to 317 MRs/month in Gnome and from 14 to 248 MRs/month in Mozilla would increase the odds by 95 and 20 percent correspondingly. Increasing the social clustering by 0.11 in Gnome and by 0.2 in Mozilla leads to 22.15 and 48 percent increase in the odds. The macro-climate environment of higher product popularity is associated with lower odds to become an an LTC—increasing the number of users

TABLE 6
Model for Gnome (125,665 Observations)

	Est	Std.Err.	x	x_{alt}	$\frac{Odds(x_{alt})}{Odds(x)}$
(Intcpt)	−4.79	0.193			
nUsr	−1.95	0.0908	0.528	0.87	51/100
RS	−0.981	0.0588	−0.794	−0.468	73/100
GotFix	0.829	0.0354	F	T	229/100
withBB	−1.08	0.0556	T	F	295/100
nCmt	0.719	0.0314	ln 2	ln 4	165/100
nPeer	−0.0543	0.00673	ln 1779	ln 5405	94/100
pShared	2.00	0.182	ln 1.06	ln 1.17	122/100
LckAttn	−0.501	0.0778	F	T	61/100
PeerPerf	0.218	0.00496	ln 15	ln 318	195/100

15. For SAS users these are so-called Type-I sum of squares where the predictors are added sequentially to the model and the deviance explained for a predictor is adjusted for the preceding predictors. It represents a standard output for R.

TABLE 7
Deviance for Gnome Model

	Df	Deviance	Percent.Dev (%)	P-value
NULL	125,664	39,302		
nUsr	1	3,759	9.56	7.6e-103
RS	1	92	0.23	1.56e-62
GotFix	1	1,580	4.02	1.77e-121
withBB	1	516	1.31	2.64e-84
nCmt	1	703	1.79	1.72e-116
nPeer	1	57	0.15	6.91e-16
pShared	1	147	0.37	3.01e-28
LckAttn	1	1	2.54e-05	1.18e-10
PeerPerf	1	1,874	4.77	0
prj	39	245	0.62	

by 34 percent in Gnome and by 26 percent in Mozilla reduces odds by 49 percent and by 15 percent respectively. Project's RS is associated with lower odds of becoming an LTC in Gnome (0.326 of RS increase leads to 27 percent decrease in the odds). Having the size of the peer group increase from the median to the third quartile is associated with a small decrease in the odds (6 percent) in Gnome and an increase (14 percent) in Mozilla. In summary:

Observation 1. The probability of a newcomer becoming an LTC is associated with her extent of involvement and environment. Her pro-community attitude manifested in her choice to start with a comment instead of a new bug report or in her use of Bugzilla instead of a crash reporter, and the amount of effort she provides to the community, are associated with the most dramatic increases. On the contrary, her macro-climate with high project popularity, and her micro-climate with low attention, reduce her odds. Meanwhile, the attributes of her peer group, in particular, its social clustering and productivity significantly influence her opportunity to become an LTC.

5.2 Predicting Who Will Become an LTC

To evaluate if it is possible to use the results in practice, i.e., to tell who will stay in the project based on the data collected during their first month of participation, we predict LTCs among participants joining Mozilla from January of 1998 until December of 2007, i.e., the participants used to fit the model. The results are shown in Table 8. Among the participants whose predicted LTC probability is above 99th percentile, 22.15 percent became LTCs (precision), and 24.6 percent of all LTCs were in the 99th percentile group (recall). Given the extremely small probability of becoming an LTC, the precision and recall are quite high. A predictor that randomly selects one percent of newcomers would have the precision of 0.9 percent or 25 times lower and a recall of 1 percent or 25 times lower. The random predictor represents an equal likelihood for every newcomer to become an LTC. A better comparison would be with the rules of thumb by which the project members may be assessing the newcomers, but we are not aware of such rules of thumb. Based on the model, a hypothetical rule of thumb may be to identify newcomers contributing many comments as potential LTCs, because comments are easily observed and the

TABLE 8
Prediction Performance

	Data	Precision	Recall	Prob. Percentile
Mozilla	Original Data	22.15%	24.6%	99
	New Data	19.52%	37.4%	99
	New Data	46.38%	24.6%	99.7
Gnome	Original Data	38.49%	10.56%	99

model shows a very large effect size for this predictor. Predictor using 10 or more comments in first month has the precision of 13.7 percent and a recall of 26.2 percent for Mozilla. It is only half as good as the model predictor, but it has 13 times higher precision than a random predictor.

In general, there is an inverse relationship between precision and recall, where it is possible to increase one at the cost of reducing the other. For a logistic regression model we simply need to change the predicted probability limit above which we classify the participant as an LTC. In particular, the above precision and recall are based on the limit of 0.08712. If we lower the limit to 0.04959, the precision decreases to 14.98 percent and the recall increases to 33.28 percent.

Prediction of Gnome newcomers for the period from January of 1999 until December of 2007 resulted in precision of 38.49 percent and recall of 10.56 percent using probability limit 0.345 (above 99th percentile), as shown in Table 8. A predictor that randomly selects one percent of newcomers would have the precision of 3.6 percent or 10 times lower and a recall of 1 percent or 11 times lower.

5.3 Checking for Publication Bias

Almost invariably, published significant relationships tend to become less significant or disappear once more data are collected [42]. We therefore, collected more data to validate the observed Mozilla relationships reported in Table 4 (we are still unable to obtain additional Gnome data).

We have done two types of validation. First, we retrieved an additional Mozilla extract (Mozilla 2012 in Table 2) to verify the published model predictions of LTCs among the 25,406 Mozilla newcomers joining between January of 2008 and April of 2009. If the prediction performance were to drop, that would mean that the original model was likely overfit. Second, we reproduce the model and report the model coefficients and the p-values for each coefficient to determine if the significant relationships hold and if the p-values have increased (as hypothesized by [42]) after the Mozilla community offered a Bugzilla dump (Mozilla 2013 in Table 2).

5.3.1 Prediction

Using predicted probability limit of 0.05108 obtained by selecting the top 1 percent (254) of these new participants the precision was 19.52 percent and the recall was 37.4 percent, as shown in Table 8. A prediction with the same recall (24.6 %) as in the original data, has an even higher precision—46.38 percent. The prediction performance on the new dataset is not worse than on the original dataset used to fit the model, suggesting that the original model was not overfit. For comparison, a predictor that randomly selects

TABLE 9
Comparing Models for Mozilla 2011 and 2013
(170,237 Observations)

Coeff	Est'11	Est'13	z-val'11	z-val'14	change
(Intcpt)	−7.49	−7.18	−17.87	−23.031	+
nUsr	−0.601	−1.09	−4.00	−8.238	+
RS	0.701	0.19	2.39	0.684	
GotFix	0.74	0.84	8.90	11.138	+
FNotRep	0.507	0.40	6.17	5.577	−
nCmt	0.819	0.73	20.02	20.857	+
nPeer	0.142	0.14	6.92	7.970	+
pShared	2.35	2.55	17.40	21.035	+
LckAttn	−0.325	−0.42	−2.62	−4.548	+
PeerPerf	0.0649	0.07	4.95	6.473	+

one percent of newcomers would have the precision of 0.516 percent or 38 times lower and a recall of 1 percent or 37 times lower.

5.3.2 Reproducing the Model

Table 9 compares the fitted values and z-scores for Mozilla based on 2013 and 2011 extracts. First, all the predictors that were significant in the original model are still significant except for RS. We should note that the p-value for RS in the original model was fairly large at 0.02, suggesting that RS may not play a role in retention of newcomers in Mozilla. In fact, as suggested by Johnson [43], a more appropriate p-value for statistical evidence is 0.005. Based on this revised standard the originally reported p-value would not have been significant. Second, p-values for the remaining coefficients either stayed similar ($FNotRep$) or have decreased (absolute z-scores increased). Both of these findings suggest that the original model was not overfit and that the new data (40 K observations or one third of the size of the original set) added further support to the model. In summary, we did not observe the reduction in significance as suggested by Ioannidis [42], but we did find support for the more stringent levels on p-values suggested by Johnson [43].

In the course of reproducing earlier results we also discovered the differences among the extracts and the effect they had on the model. First, the Mozilla 2013 extract differs from the extracts we retrieved from webpages in several ways. In particular, we have individual IDs used by MySQL database backend for Bugzilla, eliminating issues with consistency of web-based retrieval where the same ID may map to different email/name (when the name/e-mail changes in the course of retrieval). However, the dump was sanitized because of legal, human resources, privacy or security concerns. For example, the following data were removed: all non-public products and all data associated with them, all data (bugs) in security groups, all insider group comments and attachments, all security groups, and all sensitive user account data. There were about 50,950 issues missing. In general, these data are not open to public, so they would not be in the web extracts either (unless they were not considered to be sensitive at the time of the web extract).

Second, Bugzilla does not track past states for all attributes. As we discovered, QA contact is removed from the status page if that individual leaves the project. As a consequence, only 264,295 issues of the 774,810 issues in the Mozilla 2013 extract have a QA contact. However, 592,933 issues from 620,511 issues in the Mozilla 2011 extract have a QA contact. This suggests that to ensure reproducibility, one either needs to keep track of attributes that are not tracked in Bugzilla, add such tracking to Bugzilla, or base the measures only on attributes for which past values can be reproduced. Sometimes it is not clear in advance what aspects of the data may change, therefore having multiple snapshots of the data is also advisable.

We have used QA contact as the last person in the work chain of a single issue, so it did affect our work network measures. However, our model appears to be not very sensitive to the QA contact attribute.

6 Validation

All actions of participants are captured in great detail in the ITS. However, data quality is a critical concern when analyzing ITS artifacts [44]. In this section we design a semi-structured survey to validate the critical pillar of this study: that the ITS artifacts capture what actually happened to contributors and that we interpret the artifacts correctly.

The primary goal of the survey is to validate completeness and accuracy of the events recorded in the ITS. In particular, that the contributors are correctly identified, that they were the ones creating the artifacts, that the selected artifacts correctly represent their first and last activities, and that the ITS is the channel for them to start contribution in the project. Once these facts are established, the analytic construction (i.e., the measures and the model) are based on facts.

The secondary goal of the survey is to elicit respondent opinions about some of the measures and the observed relationships among the measures (the model). On one hand, we seek additional support for the higher-level interpretation of the analytically derived measures. On the other hand, we attempt to check for large discrepancies between the analytic construction based on facts (i.e., the measures and the model) and the opinions expressed by participants. Respondents may provide evidence for mechanisms or point out latent variables overlooked in the model. Discrepancies may also reveal interesting areas for further research.

We design the survey in Section 6.1. We validate the ITS events in Section 6.2, and verify if the participation measures and modeling results agree with the participants' perceptions in Section 6.3.

6.1 Conducting Survey

We conducted two rounds of survey. In the first round we tailored emails for 40 respondents and got a basic understanding of why and how people participate in FLOSS projects. In the second round we sampled 240 respondents and used the understanding from the first round to conduct the survey.

6.1.1 Survey Principles

We followed the principles provided in [45] to design the survey, and made a trade-off between the need to obtain more responses (because of the low response rate in recent online surveys [45]), and having meaningful information in each response.

Web-based survey is the most convenient form for respondents to answer, but most of the survey tools do not allow tailoring questions with individual's background, in particular, with the concrete activities they were engaged in during the first month with the project. Respondents typically have difficulty remembering what they did and how they felt long time ago [46]. It is, therefore, important to ask questions about the artifacts of the respondent's own work from that period to generate specific retrieval cues and context to trigger recall. Our main purpose is to validate the interpretation of the detailed artifacts presented to subjects, thus the issues with recall of past events are of lesser concern. Consequently, we sent emails to respondents containing questions targeting her specific activities.

During the survey, we didn't offer incentives to encourage respondents to answer the questions, and didn't use any reminder in the first round survey. However, we sent reminders up to four times with an interval of five days each in the second round survey to increase the response rate.

6.1.2 Designing Questions

In the first round survey (questions in Appendix A, available in the online supplemental material) we attempt to understand such factors as the motivation people had for joining the project and the starting channel for contribution. The responses also indicated the necessity to tailor questions to individual respondents, e.g., "... a lot of [e]mails from researchers ... but nobody ever took the time to write individual [e]mails...".

In the subsequent round of survey (questions in Appendix B, available in the online supplemental material), we presented even more extensive artifacts of respondent's past activities. This was done to reduce the need to rely on respondents' (often fallible) memories and to stimulate their interest to respond. We separated the survey into two parts to achieve ease of response and customization. In one part we designed questions that present artifacts and ask specific questions about respondent's work. The second part includes general questions that applied to all participants. For example, to validate our understanding of common practices, we asked if the respondent believes that newcomers usually start contributing via discussion group, Bugzilla, or code. We also asked questions to validate if the LTC predictors in the model match her perceptions of what helps to retain contributors.

Most of our survey questions were close-ended with a few optional open-ended questions for collecting participants' "insights" and "experiences".

Before publishing the survey and making it publicly available, we asked four external people—one senior PhD student and three experienced developers—to review the survey in order to make sure that all questions were appropriate and easily comprehended. The responses helped us to refine the questions by eliminating questions that respondents had difficulty answering.

6.1.3 Sampling Participants

We employed stratified approach to sample contributors [34]. In the first-round survey, we randomly sampled 10 LTCs and 10 non-LTCs from each project (using $sample$ function of R [47]).

In the second round, we stratified LTC and non-LTC groups using three factors that explained a substantial amount of prediction variance: $GotFix$ (representing the extent of involvement), $nUsr$ (representing macro-climate), and $LckAttn$ (representing micro-climate). $GotFix$ and $LckAttn$ are boolean parameters, and we sampled $nUsr$ from the top 20 percent and from the bottom 20 percent. Our objective was to get 10 responses from LTCs and 10 responses from non-LTCs from each project. The first-round survey shows that the response rate and the proportion of delivered emails was much higher for LTCs. Based on that estimate, we sampled five participants from each LTC group and ten from each non-LTC group. Our sample had 120 logins from each project: 40 LTCs and 80 non-LTCs.

There is no overlap between the first round respondents and the second round respondents.

Finally, we picked four senior project participants (two from each project) and asked them to comment on our understanding of their project practices and on our findings about newcomers' activities and their retention. We also included general survey questions.

6.1.4 Response Rate

Out of 40 emails we sent in the first round, eight were not delivered, and seven responded, giving us the response rate of 22 percent. Out of 240 emails we sent in the second round, 71 could not be delivered, and 29 responses were usable for our analysis (the response rate of 17 percent).

There is one response that we considered unusable and dropped from the analysis, the respondent responded to our email and claimed he wouldn't answer the questions because, "Given that you call GNOME an OSS project, i don't think I want to participate. GNOME is a free software project." All four senior participants responded. In the later analysis, we refer to them as Respondent 1 through 33.

6.2 Validating ITS Events

Validating completeness and accuracy of the events recorded in the ITS is the primary purpose of the survey because it ensures that individuals are accurately identified, that they were the ones creating the artifacts, that the selected artifacts correctly represent their first and last activities, and that the ITS is a channel to start contribution in the project.

The respondents agreed with the basic facts about themselves that we retrieved from ITS data and used in our models. Many started their contribution from ITS and became LTCs later (some may also become long term code contributors), providing evidence that ITS is an important channel to recruit valuable contributors in FLOSS projects. Only one respondent disagreed with the definition of LTC.

In particular, 24 out of 29 (83 percent) respondents (the other four respondents were senior members whom we did not ask about their own activities in the first month) agreed that our artifacts documenting their first and last activities in Mozilla or Gnome were consistent with the facts. Three of the remaining five noted that they were not able to confirm or deny, because, e.g., "this bug was submitted over ten years ago and I do not recall any specifics about my experience". Two disagreed with our artifacts about their

TABLE 10
How to Start Contributing

Channel	Bugzilla	Mailing lists and forums	Both	Don't know	Other
# of Responds	3	12	7	5	2

TABLE 11
Categories of Predictors

Category	Description
TASK	Statement about the number or type of *specific* tasks and activities clearly related to the project that the respondent or some other *specific* person has done, is capable of doing, or likes doing.
ISSUE FIX	Statement about the explicit or implicit emotions the respondent or somebody else had towards bugs (not feature requests): the desire not to have/eliminate them, pride or satisfaction of finding, fixing, or responding to them, the frustrations with the process needed to resolve them.
PEERS	Explicit statement about the relationship between people (respondent and her peers or somebody and her peers) that may provide or prevent people's opportunity to contribute or to contribute in new ways: what people do may affect others or be affected by others.

first activity in the project, in particular, one respondent noted: "I believe my first interaction with the community was through a newsgroup", while the other noted "I was the initial developer of xxx long before that" (xxx is the name of a Mozilla sub-project).

All but one of the respondents didn't object to being referred to as LTC or non-LTC. The only reservation was: "I'm in no way a "long term contributor"". She has been working for a company that provides professional Embedded Linux services for industrial customers, and helped with ten bugs over three years (from Aug 2007 to Aug 2010). Three bugs per year may not appear productive, but she is among the top 90 percent contributors (measured by number of bugs she modified a year) who stayed for at least three years in Gnome.

In this study we only considered contributors joining Bugzilla, thus we may have missed contributors who started by making code commits or by participating in a project-related forum. Even though 83 percent of our respondents confirmed that they have started by contributing to Bugzilla, their opinions varied about which channel a typical newcomer would use. Table 10 shows that, 12 respondents believed newcomers usually start from mailing lists and forums, 3 chose Bugzilla, 7 chose both, and nobody chose code commits. Surprisingly, among people who chose mailing lists and forums, two claimed that their personal preferences are different, in particular, one said, "I personally dislike forums, because most feedback is of very low quality, so I usually start with a bugtracker to have some idea of how responsive the team or community is" (Respondent 2); the other said "(it) Depends on what technical background they have", "Many people start contributing using chat/forums. Other start coding right away etc. Bugzilla is for the tech wizards" (Respondent 15).

The seven respondents who selected either "Don't know" or "Other" for the contributor starting channel appear to have a different definition of "what counts as a contribution" and feel that "the type of contributor" matters. For example, Respondent 7 said: "Depends on the contribution area, only coders would start in Bugzilla, the majority probably start in option C (i.e., mailing lists and forums)"; Respondent 8 claimed: "I don't really know, depends on the type of contributor and respective reasons to contribute". In other words, the starting channel may depend on the type of the contributor. In fact, six respondents (among 29) mentioned that the variation among contributors may lead to different types of involvement. In particular, individuals with technical background, or simply, "coders" are believed to get involved through ITS, because "It's the public facing way to get involved. Bug reports are easy to submit and verify and are objective" (Respondent 4).

To conclude, the responses provide evidence that the ITS data used in this study captured the contributors' initial involvement in the project. Moreover, the contributors starting from ITS tend to have a more technical background (83 percent respondents started from ITS, 34 percent respondents believed newcomers may start from ITS, and 21 percent respondents believed coders are more likely to start from ITS), suggesting they may be more likely to become valuable code contributors than participants starting from, e.g., forums. Note that 25 out of 29 (86 percent) respondents told they were volunteers when they started contribution.

6.3 Feedback from Participants

With the assurance of basic facts as described in last section, the measures and the model that we constructed are established as reflection of facts. How is that perceived by respondents may provide evidences for our results and inspire future research.

6.3.1 *Categories of Predictors*

The answers to the questions in the first round contained significant amount of information regarding motivation of participants, reasons they started/stopped contributing, and the role of their environment. We tap this rich source of information to interpret our measures of involvement and environment. In particular, we developed a schema to categorize the sentences in the responses with respect to tasks and issue fixes to interpret these two measures of involvement and with respect to peers to interpret micro-climate. Two authors created category definitions together (shown in Table 11) and followed them to assign sentences to categories independently. Cross-rater agreement had Cohens Kappa above 73 percent, indicating good agreement.

6.3.2 *Agreement from Respondents*

Analysis of the survey and interview responses provides some support that model predictors are meaningful and that they discriminate LTCs from non-LTCs.

First, the categorization of the open-ended responses suggests that task-, issue fix-, and interaction with peers-

TABLE 12
Categorizing Answers to the Open-Ended Questions

Category	Answers to Open-ended Questions
TASK	*I participate on documentation and website bugs for Gnucash, as it is the best use of my abilities. And I hope by doing this, some developer will consider fixing my bugs.* *I just like testing software (in this case nightly/firefox builds) and sometimes come across bugs. Probably most important factor was/is the fact i can download nightly/hourly builds and can see/interact with the developers/coders in the Bugzilla.* *I don't become LTC because I don't have enough time/knowledge to resolve issues in Mozilla by myself (it is too complex for me).*
ISSUE FIX	*If you have faced a bug, you need to make some effort to describe it. Then you must check if there is duplicates. Then you create report and wait until response. All time you are waiting you must keep an issue in mind. After initial response there is good possibility that devs (developers) can't or don't want to reproduce the issue and you must know how to diagnostics and how to prove that issue is really exists. Then you wait until issue is fixed usually without any feedback on progress. When issue gets fixed you should confirm this, but you waited for a long time and probably forgot some details. Also where to get fixed binaries? They were released or you must compile them from sources for your platform?* *If I don't annoy developers constantly with the issue or don't fix it by myself it wont be fixed.*
PEERS	*I learned a lot from this leading open source project while working with other brilliant contributors.* *I felt an obligation to help support the project and the people who were associated with it, many of whom were my coworkers at Netscape.*

related topics were noted by contributors. We tagged 98 sentences and for each of the three categories selected several quotes shown in Table 12. Of them, 25 sentences related to issue fix, 20 to peers, 24 to tasks.

In particular, quotes from $TASK$ category suggest that participants are aware of the type of tasks they are capable of (and like) doing and the value they provide, and this may have helped them to engage with the project.

The amount of effort spent on getting reports fixed (category $ISSUEFIX$) appears to be a good indicator of the extent of involvement. The two quotes presented in Table 12 articulate reasons that make it difficult to fix a bug. In addition, the quotes suggest challenges attracting developer attention "[I had to] annoy developers constantly." A leading contributor confronted with this comment admitted the likelihood of such situation.

Meanwhile, the interaction with the peers was often quoted (20 percent of the sentences). For example, Table 12 suggests that learning from productive peers and attachment to co-workers are two concrete manifestations of micro-climate.

Second, the responses to close-ended questions helped to quantify the respondents perception on how the individuals' initial activities and interactions affect the chances to stay with the project.

As described in Section 6.1, we shortened the list of survey questions to make it easier to answer (and, thus, get a higher response rate). We chose five representative measures to solicit respondent agreement with the findings provided by the model. The measures were: $GotFix$, representing extent of involvement; $nUsr$ and RS, representing macro-climate; and $nPeer$ and $LckAtten$, representing micro-climate. We used five-point scale with anchors ranging from "strongly agree" to "strongly disagree" (i.e., the agreement score is from 2 to -2). Table 13 lists the frequency of agreement scores for the four factors, for example, the frequency of -2 is zero for all four factors, suggesting nobody chose "strongly disagree". It also shows that the agreement scores for the factors were significantly greater than zero with a p-value less than 0.002 (t-test), suggesting the respondents' agreement.

TABLE 13
Responses: Agreement with Model

Agreement Level/Factor	GotFix	nUsr	RS	nPeer
2	11	6	8	3
1	14	10	15	12
0	2	10	1	6
-1	1	2	2	3
-2	0	0	0	0
p-value	2.04e-09	2.50e-04	2.64e-07	1.94e-03

To validate $LckAtten$ predictor we presented the respondent with an issue she reported during her first month that had the slowest response time and asked if she perceived that response time to be too fast, normal, too slow, other, or "don't know". Of the 23 respondents who answered this question, 13 chose "normal", five chose "other". The two respondents who chose "Too slow and, thus discouraging" were non-LTCs, while the remaining three chose "don't know".

Some of respondents couldn't remember what happened that long ago. For example, Respondent 3 stated: "I cannot quantify how I "felt" that long ago", therefore he chose "other". Respondents made their choice mostly based on how they perceived the reported delay at the time they answered the question. For example, Respondent 1 explicitly explained her choice (choosing Normal): "I think this reply time would be normal in OSS community". Respondent 23 noted: "First response in 14 hours or 14 days is OK."

Meanwhile, the four senior players believed that response time is associated with the chances that a newcomer will contribute for a long time. Two of them chose "strongly agree", the other two chose "agree". Ironically, we did not get a strong support that response time matters from the individuals confronted with their own issues, but the community leaders felt it to be an important factor.

7 Limitations

We discuss some of the limitations related to the ITS data and analytic constructs in Section 7.1, and specific issues encountered while identifying Bugzilla participants in Section 7.2. The internal and external validity are presented in Section 7.3.

7.1 Limitations of Bugzilla for Mozilla and Gnome

We start from the consistency and accessibility of the issue tracking data in the Bugzilla of Mozilla and Gnome.

First, to retrieve Bugzilla data we obtained all issues from 1 to the largest number, e.g., 645,899 for Mozilla 2011 extract and 639,379 for Gnome 2011 extract. Some issues were either not public or not obtainable, e.g., 121,578 in Gnome 2011 and 25,388 in Mozilla 2011. We compared different snapshots extracted at different times, and verified to make sure that the later snapshots include the issues in the earlier ones. We had no problems obtaining Mozilla data, in fact Mozilla graciously provided the entire dump of the Bugzilla. We encountered a number of problems with Gnome. The policy of Gnome Bugzilla is to prevent the retrieval of large numbers of issues with complete email address of contributors. We, therefore, had to rely on public extracts of Gnome Bugzilla data (i.e., Gnome 2006 extract and Gnome 2008 extract) and, instead of full email address, we had only a login which we had to map to an individual.

Second, the data might not reflect what actually happened, e.g., Gnome Bug 572011 doesn't have an information page or an xml file, but it has an activity history page. Some MRs have some states missing. For example, for Mozilla Bug 235354 the resolution type on the information page was "Status: RESOLVED NOTABUG", but on the history activity page the last resolution was INVALID. To address this limitation we tested how sensitive our analysis results are to these data consistency issues. Nine-hundred ninety five MRs in Gnome and 601 MRs in Mozilla had intermediates states missing, but excluding them didn't have any noticeable impact on the results.

We continue with the sensibleness of the measures we construct: do they reflect the intended concepts? Individuals and their environment are notoriously difficult to measure because of the variability among individuals and the ambiguity of concepts such as the extent of involvement and environment. The issue tracking systems, however, record the details of the activities individuals engage in, thus providing an opportunity to infer the effort they spend to accomplish them. These basic measures then provide a basis to estimate their extent of involvement and their environment.

The subjects and their roles present another limitation to our analysis. We investigated the participants' activities and their interaction with environment when they joined the project. However, some participants may join as developers instead of newcomers, i.e., commit a change to the code repository before leaving a trace in Bugzilla. However, only one person from the 29 survey respondents ($1/29 = 3$ percent) appears to have been in the development team from the very beginning. This suggests that such occurrences are rare in practice.

7.2 Issues with Participant's Identity

Bugzilla records do not provide sufficient information to identify individuals without uncertainty. We, therefore, compared several approximate identifications that are based on email, login, and full name.

7.2.1 Email-, Login- and Name-Based Identification

A single participant may have multiple emails and/or logins, e.g., 10 percent of the full names are associated with at least two different emails in Mozilla 2011 extract. A participant who changes her email after joining the project would be considered as several different participants if email is used as a proxy for a person. On the other hand, a single e-mail or login may be used by multiple participants. For example, it is common for certain roles, such as QA, to share the same email.

The same login may represent different emails, possibly of different participants. For example, emails zhmh@pku.edu.cn and zhmh@avaya.com share the same login zhmh, but they may represent different individuals. Therefore two different non-LTCs with the same login and participating three years apart may be identified as a single LTC. Our survey sample of Gnome demonstrated this issue: 12 LTCs from 40 sampled LTCs were, in fact, two distinct individuals who shared login.

A participant may be identified by the associated name and all emails/logins used for a particular name may be linked to a single person. This approach suffers from some of the issues identified above: the same name may be used by multiple people or the same person may change her name. Furthermore, not every login/email is accompanied by a name, so for some participants only email (or login) is available.

7.2.2 Sensitivity of Results

To test the sensitivity of the model to the choice of method used to identify an individual we first identified and removed administrative logins, such as, "mozilla", "gnome", "Bugzilla*", and "*maint" because we focused on ordinary contributors and because multiple individuals were sharing such logins. We then used the full name of the participant associated with each login/email to identify all multi-name IDs and multi-ID names. To determine if the particular way of identifying an individual affects our modeling results we fit the models on three datasets: the original data (ID being email in Mozilla and login in Gnome), the data excluding administrative IDs, and data using name as the ID (when name was available, otherwise using original non-administrative ID). The results on all three datasets were similar and we reported the results using original data.

7.2.3 Inconsistencies between Bugzilla Extracts

In theory, all the individuals identified in the Mozilla 2011 extract should also be in the Mozilla 2012 extract, because the later extract is an extension of the earlier extract with additional issues reported since 2011 and additional history of the issues modified since 2011. However, 459 emails in 2011 extract were missing from 2012 extract.

This was partly caused by the changes of email attribute associated with individual's key in the Bugzilla's MySQL database. Because our data are based on the issue summary and history web reports, the email (attribute) of the same participant may be different for issue reports generated at different times. We inspected occasions in which the same activities in the two extracts were associated with different emails, e.g., Bug 490556 reporter is associated with *brent@aerobrent.com* in the 2012 extract and with *aerobrent@gmail.com* in the 2011 extract, and found 439 out of 459 emails were linked this way. To resolve these inconsistencies we first obtained emails present in the 2011 extract but not present in the 2012 extract; Second, for each missing email, we located the associated activity in the 2011 extract; Third, we located the same activity in the 2012 extract using the issue number, the type of activity, the order of the activity, and the issue attributes that were changed. We used actor's email in the corresponding activity to identify the change of email and associated both emails with the same individual.

We also discovered 17 of missing emails were mentioned in Bug 452498. A further investigation revealed an error (which we fixed) that got introduced either generating or retrieving the issue report. In Mozilla 2011 extract, all the activities related to Bug 452498 had only a login instead of the full email. For example, instead of *shaver@mozilla.org* there was just *shaver*. This suggests that the retrieval script had not successfully logged in into Bugzilla before retrieving the issue (Bugzilla reports for unauthenticated users do not include a full email address).

The rest three missing emails were due to the change of email while we were obtaining the 2011 extract. For example, a person changed email while we were retrieving the issues (the single extract took approximately two weeks to complete), so that we get one email for the issues retrieved before the change and another email for the remaining issues. It poses even more problems when comparing or merging two extracts. For example, in the 2011 extract, *Jason Duell* has *jduell@alumni.princeton.edu* associated with his activities in one issue report, but has *jduell.mcbugs@gmail.com* associated with the remaining activities in other reports. To confirm that it was the database change, we verified in the 2012 extract that *jduell@alumni.princeton.edu* was replaced by *jduell.mcbugs@gmail.com*.

7.3 Internal and External Validity

From the internal validity perspective we checked the assumptions for the logistic regression. We also log-transformed the predictors to make the model coefficients more interpretable and to reduce the influence of the potential outliers. While only $19\ \text{percent}(23\ \text{percent})$ of the deviance is explained by the model, this is, in fact, an excellent fit given that only $0.9\ \text{percent}(3.6\ \text{percent})$ of the participants become LTCs in Mozilla (Gnome). Typically, the more imbalanced the response is in the logistic regression, the lower the fraction of deviance can be explained by the model.

The model can also be evaluated by how well it predicts LTCs. The top 1 percent (by predicted probability) of 25,406 Mozilla participants in the validation set, 19.52 percent were LTCs (precision) and 37.4 percent of all LTCs were in that top 1 percent (recall). For comparison, a predictor randomly selecting 1 percent of the participants has precision of 0.516 percent or 38 times lower, and recall of 1 percent or 37 times lower.

The way Mozilla and Gnome are operating is not unusual for a FLOSS project, thus not a threat to external validity. The models for both projects are quite similar, thus there are no reasons to expect that other projects would differ. However, both are large projects and both represent user interface domain. We, therefore, may not generalize to other domains (e.g., server), and smaller projects.

It's important to stress that our findings show association between the response and predictors, but that association may not be causal. In particular, there may be some aspects of individual character or of the environment that we did not measure, but that cause both the response and the predictors to behave in the observed pattern.

8 Conclusions

In this study, we address the following question: what impacts the chances that an ITS participant will become an LTC? Is that probability related to participant's character, project's climate, or the interaction between the participant and her environment? We proposed several measures of ITS participant behavior using issue workflow and modeled how the variations in their behavior are associated with the probability of the participant becoming an LTC.

We found the probability of a newcomer becoming an LTC to be associated with person's extent of involvement and interactions with her environment. Most importantly, measures of her pro-community attitude represented by starting contributions with activity other than reporting an issue and by reporting an issue via Bugzilla instead of a crash-reporter, double her odds of becoming an LTC. The ability to have at least one issue reported during her first month to be fixed, also doubles her odds. The micro-climate of low attention represented by a too-rapid response to issue reports, and macro-climate of high project popularity reduce her odds. Individual's peer network size had opposite effects in the two projects and project's relative sociality was significant only in Gnome. This may reflect some inherent differences between practices of Mozilla and Gnome that need further study.

From the methodological perspective we discovered that to ensure reproducibility of the results it is essential to base measures on attributes that can be reconstructed either via internal change tracking of ITS or by retrieving several snapshots of ITS data. Based on the results of replicating our analysis, we also found support for more stringent limits for statistical significance proposed by Johnson [43].

In summary, the main contributions of this study include:

- The ITS-based measures of participants' initial behavior (extent of involvement and interaction with environment);
- A model relating the extent of involvement and macro- and micro-climate to the odds of becoming an LTC;
- A practical method to predict who will become an LTC based on the initial actions and environment of a participant.

The findings may help individual participants to interpret their experiences in the project, understand project needs, and find the best ways to contribute. It may also help FLOSS communities to adopt better strategies to attract and retain newcomers and help project members understand the likely return on their investment in supporting newcomers. For example, GotFix was found to double the chance of a participant becoming an LTC. The community, therefore, may consider devoting their limited time and effort to these potential future LTCs. Ironically, it is during the times when projects are most popular and streams of new participants are overwhelming the mentors, the community needs to put extra effort to retain newcomers.

We communicated with the leading players in Mozilla and Gnome communities both to obtain feedback and to act on our findings.[16] This resulted in Mozilla providing bugzilla data for researchers.[17] In the words of the head of Mozilla's community building team: "*I think the ways that people originally got involved with the project haven't scaled well as we've gotten bigger. The original organic and informal processes worked when we were small, but the barrier to entry has gotten too high as we've grown so much bigger.*" Therefore, "*I believe that we've needed to replace the old model with a new model that includes providing explicit pathways that people can follow and by helping make connections between new contributors and existing community members who can act as mentors.*"

Acknowledgments

Partial support by the National Basic Research Program of China Grant 2015CB352200, the National Natural Science Foundation of China Grants 91118004 and 61432001, and the National Hi-Tech Research and Development Program of China (863) Grant 2012AA011202. The authors thank Andre Klapper, Johannes Schmid, David Boswell, Thomas Düllmann, Enrico Nicoletto, Andrey Bondarenko, and many others who participated in this study. They thank the reviewers for their many valuable suggestions that greatly improved the paper.

References

[1] A. Mockus, R. T. Fielding, and J. Herbsleb, "Two case studies of open source software development: Apache and Mozilla," *ACM Trans. Softw. Eng. Methodol.*, vol. 11, no. 3, pp. 1–38, Jul. 2002.

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

[3] K. R. Lakhani and E. von Hippel, "How open source software works: Free user-to-user assistance," *Res. Policy*, vol. 32, no. 6, pp. 923–943, 2003.

[4] E. von Hippel and G. von Krogh, "Open source software and the private-collective innovation model: Issues for organization science," *Org. Sci.*, vol. 14, no. 2, pp. 209–223, Mar./Apr. 2003.

[5] J. Anvik and G. C. Murphy, "Reducing the effort of bug report triage: Recommenders for development-oriented decisions," *ACM Trans. Softw. Eng. Methodol.*, vol. 20, no. 3, pp. 10:1–10:35, Aug. 2011.

[6] J. Xie, M. Zhou, and A. Mockus, "Impact of triage: A study of Mozilla and Gnome," in *Proc. ACM/IEEE Int. Symp. Empirical Softw. Eng. Meas.*, Baltimore, Maryland, USA, Oct. 10–11, 2013, pp. 247–250.

[7] W. Scacchi, "Understanding the requirements for developing open source software systems," *IEEE Softw.*, vol. 149, no. 1, pp. 24–39, Feb. 2002.

[8] C. R. Reis and R. P. de Mattos Fortes, "An overview of the software engineering process and tools in the Mozilla project," in *Proc. Open Source Softw. Develop. Workshop*, 2002, pp. 155–175.

[9] K. Schmidt and C. Simone, "Coordination mechanisms: Towards a conceptual foundation of CSCW systems design," *J. Collaborative Comput.*, vol. 5, pp. 155–200, 1996.

[10] M. Zhou and A. Mockus, "Does the initial environment impact the future of developers?" in *Proc. 33rd Int. Conf. Softw. Eng.*, Honolulu, HI, USA, May 21–28, 2011, pp. 271–280.

[11] M. S. Zanetti, I. Scholtes, C. J. Tessone, and F. Schweitzer, "The rise and fall of a central contributor: Dynamics of social organization and performance in the gentoo community," *CoRR*, vol. abs/1302.7191, 2013.

[12] I. Steinmacher, I. Wiese, A. Chaves, and M. Gerosa, "Why do newcomers abandon open source software projects?" in *Proc. 6th Int. Workshop Cooperative Human Aspects Softw. Eng.*, May 2013, pp. 25–32.

[13] 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.

[14] M. Zhou and A. Mockus, "Developer fluency: Achieving true mastery in software projects," in *Proc. 18th ACM SIGSOFT Int. Symp. Found. Softw. Eng.*, Santa Fe, NM, USA, Nov. 7–11, 2010, pp. 137–146.

[15] M. Blumberg and C. D. Pringle, "The missing opportunity in organizational research: Some implications for a theory of work performance," *Acad. Manage. Rev.*, vol. 7, no. 4, pp. pp. 560–569, 1982.

[16] R. H. Rasch and H. L. Tosi, "Factors affecting software developers' performance: An integrated approach," *MIS Quart.*, vol. 16, no. 3, pp. 395–413, Sep. 1992.

[17] G. Hertel, S. Niedner, and S. Herrmann, "Motivation of software developers in open source projects: An internet-based survey of contributors to the Linux kernel," *Res. Policy*, vol. 32, no. 7, pp. 1159–1177, 2003.

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

[19] J. A. Roberts, I.-H. Hann, and S. A. Slaughter, "Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the apache projects," *Manage. Sci.*, vol. 52, no. 7, pp. 984–999, Jul. 2006.

[20] J. Koh, Y.-G. Kim, B. Butler, and G.-W. Bock, "Encouraging participation in virtual communities," *Commun. ACM*, vol. 50, no. 2, pp. 68–73, Feb. 2007.

[21] J. Hahn, J. Y. Moon, and C. Zhang, "Emergence of new project teams from open source software developer networks: Impact of prior collaboration ties," *Inf. Syst. Res.*, vol. 19, no. 3, pp. 369–391, Sep. 2008.

[22] K. J. Stewart and S. Gosain, "The impact of ideology on effectiveness in open source software development teams," *MIS Quart.*, vol. 30, no. 2, pp. pp. 291–314, 2006.

[23] S. Daniel, L. Maruping, M. Cataldo, and J. Herbsleb, "When cultures clash: Participation in open source communities and its implications for organizational commitment," presented at the Int. Conf. Information Systems, Shanghai, China, Dec. 7 2011, p. 7.

[24] W. F. Boh, S. A. Slaughter, and J. A. Espinosa, "Learning from experience in software development: A multilevel analysis," *Manage. Sci.*, vol. 53, no. 8, pp. 1315–1331, Aug. 2007.

[25] K. Ehrlich and M. Cataldo, "All-for-one and one-for-all?: A multilevel analysis of communication patterns and individual performance in geographically distributed software development," in *Proc. ACM Conf. Comput. Supported Cooperative Work*, 2012, pp. 945–954.

[26] S. O'Mahony and F. Ferraro, "The emergence of governance in an open source community," *Acad. Manage. J.*, vol. 50, no. 5, pp. 1079–1106, Oct. 1, 2007.

[27] R. E. Kraut and P. Resnick, *Building Successful Online Communities: Evidence-Based Social Design*. Cambridge, MA, USA: MIT Press, 2012.

[28] N. Ducheneaut, "Socialization in an open source software community: A socio-technical analysis," *J. Comput. Supported Collaborative Work*, vol. 32, pp. 323–368, 2005.

[29] B. J. Hoffman and D. J. Woehr, "A quantitative review of the relationship between personorganization fit and behavioral outcomes," *J. Vocational Behav.*, vol. 68, no. 3, pp. 389–399, 2006.

[30] A. Mockus, "Software support tools and experimental work," in *Proc. Int. Workshop Empirical Softw. Eng. Issues: Critical Assessments Future Directions*, 2007, pp. 91–99.

[31] J. Xie, Q. Zheng, M. Zhou, and A. Mockus, "Product assignment recommender," in *Proc. 36th Int. Conf. Softw. Eng.*, Hyderabad, India, Jun. 1–6, 2014, pp. 556–559.

16. https://bugzilla.mozilla.org/show_bug.cgi?id=838947, https://bugzilla.gnome.org/show_bug.cgi?id=693295

17. https://bugzilla.mozilla.org/page.cgi?id=researchers.html.

[32] D. M. German, "The Gnome project: A case study of open source, global software development," *Softw. Process: Improvement Practice*, vol. 8, no. 4, pp. 201–215, 2003.

[33] P. A. Wagstrom, "Vertical interaction in open software engineering communities," Ph.D. dissertation, Carnegie Institute of Technology and School of Computer Science, Carnegie Mellon Univ., Pittsburgh, PA, USA, vol. CMU-ISR-09-103, Mar. 2009.

[34] S. L. Lohr, *Sampling: Design and Analysis: Stratefied Sampling*. Boston, MA, USA: Cengage Learning, 2010.

[35] K. Nakakoji, Y. Yamamoto, Y. Nishinaka, K. Kishida, and Y. Ye, "Evolution patterns of open-source software systems and communities," in *Proc. Int. Workshop Principles Softw. Evolution*, Orlando, FL, USA, May 19–20, 2002, pp. 76–85.

[36] K. A. Ericsson, R. T. Krampe, and C. Tesch-Rmer, "The role of deliberate practice in the acquisition of expert performance," *Psychol. Rev.*, vol. 100, no. 3, pp. 363–406, Jul. 1993.

[37] A. L. Kristof-Brown, R. D. Zimmerman, and E. C. Johnson, "Consequences of individuals' fit at work: A meta-analysis of person job, person organization, person group, and person supervisor fit," *Personnel Psychol.*, vol. 58, no. 2, pp. 281–342, 2005.

[38] M. L. Verquer, T. A. Beehr, and S. H. Wagner, "A meta-analysis of relations between person organization fit and work attitudes," *J. Vocational Behav.*, vol. 63, no. 3, pp. 473–489, 2003.

[39] A. Ko, R. DeLine, and G. Venolia, "Information needs in collocated software development teams," in *Proc. 29th Int. Conf. Softw. Eng.*, May 20–26, 2007, pp. 344–353.

[40] A. Mockus, "Succession: Measuring transfer of code and developer productivity," in *Proc. Int. Conf. Softw. Eng.*, May 12–22, 2009, pp. 67–77.

[41] B. C. Anda, D. I. Sjøberg, and A. Mockus, "Variability and reproducibility in software engineering: A study of four companies that developed the same system," *IEEE Trans. Softw. Eng.*, vol. 35, no. 3, pp. 407–429, May/Jun. 2009.

[42] J. P. A. Ioannidis, "Why most published research findings are false," *PLoS Med*, vol. 2, no. 8, p. e124, Aug. 30, 2005.

[43] V. E. Johnson, "Revised standards for statistical evidence," *Proc. Nat. Acad. Sci. USA*, vol. 110, no. 48, pp. 19313–19317, 2013.

[44] A. Mockus, "Engineering big data solutions," in *Proc. Future Softw. Eng.*, Hyderabad, India, Jun. 1–6, 2014.

[45] D. A. Dillman, J. D. Smyth, and L. M. Christian, *Internet, Mail, and Mixed-Mode Surveys: The Tailored Design Method*, 3rd ed. New York, NY, USA: Wiley, 2009.

[46] R. Tourangeau, L. J. Rips, and K. Rasinski, *The Psychology of Survey Response*. Cambridge, U.K.: Cambridge Univ. Press, 2000.

[47] R Development Core Team, *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statist. Comput., 2008.

Minghui Zhou received the BS, MS, and PhD degrees in computer science from National University of Defense Technology in 1995, 1999, and 2002, respectively. She is an associate professor in Software Institute, School of Electronics Engineering and Computer Science, Peking University. She is interested in digital sociology, i.e., understanding the relationships among people, project culture, and software product through mining the repositories of software projects. She is a member of the ACM.

Audris Mockus received the BS degree in applied mathematics from the Moscow Institute of Physics and Technology in 1988, the MS degree in 1991, and the PhD degree in statistics from Carnegie Mellon University in 1994. He is a Harlan Mills Chair Professor of Digital Archeology in the Department of Electrical Engineering and Computer Science, the University of Tennessee. He also continues to work part-time at Avaya Labs Research. Previously, he was in the Software Production Research Department at Bell Labs. He studies software developers' culture and behavior through the recovery, documentation, and analysis of digital remains. These digital traces reflect projections of collective and individual activity. He reconstructs the reality from these projections by designing data mining methods to summarize and augment these digital traces, interactive visualization techniques to inspect, present, and control the behavior of teams and individuals, and statistical models and optimization techniques to understand the nature of individual and collective behavior. He is a member of the IEEE and ACM.