What Make Long Term Contributors: Willingness and Opportunity in OSS Community

Minghui Zhou
School of Electronics Engineering and Computer Science, Peking University
Key Laboratory of High Confidence Software Technologies, Ministry of Education
Beijing 100871, China
zhmh@pku.edu.cn

Audris Mockus
Avaya Labs Research
233 Mt Airy Rd, Basking Ridge, NJ
audris@avaya.com

***Abstract*—To survive and succeed, software projects need to attract and retain contributors. We model the individual's chances to become a valuable contributor through her capacity, willingness, and the opportunity to contribute at the time of joining. Using issue tracking data of Mozilla and Gnome, we find that the probability for a new joiner to become a Long Term Contributor (LTC) is associated with her willingness and environment. Specifically, during their first month, future LTCs tend to be more active and show more community-oriented attitude than other joiners. Joiners who start by commenting on instead of reporting an issue or ones who succeed to get at least one reported issue to be fixed, more than double their odds of becoming an LTC. The micro-climate with a productive and clustered peer group increases the odds. On the contrary, the macro-climate with high project popularity and the micro-climate with low attention from peers reduce the odds. This implies that the interaction between individual's attitude and project's climate are associated with the odds that an individual would become a valuable contributor or disengage from the project. Our findings may provide a basis for empirical approaches to design a better community architecture and to improve the experience of contributors.**

***Keywords*-Long Term Contributor; open source; willingness; opportunity; interaction of person and environment**

I. Introduction

Ancient strategist Sun Tzu claimed that to succeed in battle it is crucial to gain people's support. The same applies to software projects, because their success in the marketplace is determined by their ability to attract the support from contributors. *"OSS doesn't work without contributions from the community"*, as an interviewee for this study commented. Therefore, *"An open source community's volunteer contributors should be a treasured resource"*[1].

Understanding how to maintain, sustain, and grow the contributor community is critical for the survival and success of any open source project. For example, Wikipedia introduced *Love* button to make it easy for peers to *"encourage new editors to stay and to contribute despite the fact that their contributions often get deleted and derided"*[2]. In particular, Long Term Contributors (LTCs) are crucial for project's success, because it takes a long time for developers to become productive and even more time to be able to do central tasks, e.g., to mentor newcomers [1]. Furthermore, individual and group job enrichment efforts may be more successful when attention is directed to initial task experiences [2]. Consequently, the understanding of what affects the chances that a new contributor will become an LTC would, therefore, facilitate projects' success by helping projects train and retain developers capable of solving critical tasks.

Borrowing a framework from management science[3], which uses three interactive dimensions of capacity (e.g., ability), willingness (e.g., attitude), and opportunity (e.g., environment) to account for the work performance of individuals, we model the probability that a new joiner would become a valuable contributor (an LTC)[3] through her willingness and opportunity to contribute at the time of joining. Issue tracking systems record the history of how people initiate and complete various tasks in software projects. We, therefore, assume that such detailed data would contain traces that reflect people's ability, attitude, and environment and use issue tracking data to model willingness and opportunity in two OSS projects – Gnome and Mozilla. We gathered and inspected artifacts recorded in the bug tracking system, in historic snapshots of projects' web pages, and in published literature. We used these sources to model a participant's willingness and capacity through the activities she is willing and competent to take on (the number and type of tasks she starts with) and the effort she is willing to provide (the value of her activities to the product and community, e.g., the fraction of reported issues that were ultimately fixed). We model opportunity through environment's macro-climate shared among all participants and micro-climate unique for a person. In particular, project's status such as popularity and relative sociality (RS) [4] con-

[1]http://eaves.ca/2011/04/07/developing-community-management-metrics-and-tools-for-mozilla/

[2]http://idle.slashdot.org/story/11/06/27/1231259/Wikipedia-Adds-WikiLove-For-Newbie-Editors

[3]A participant who stays with the project for at least three years and is productive (above 10-th percentile of changes per year).

[4]A geometric average over all project's participants of the ratio of the number of individual's workflow peers (social aspect) to the number of tasks that individual participates in (technical aspect) [4].

stitute measures of macro-climate, while the initial size of peer group and actions of peers (e.g., their performance, their social clustering, or their attention to her) represent micro-climate. We fit a logistic regression model [5] to quantify the relationships between these predictors and the probability that a new contributor would become an LTC. The results show the probability to be associated with the contributor's attitude and environment. Specifically, at the time of joining, future LTCs tend to take more active role and show more community-oriented attitude than other joiners. They also receive more attention from the community and encounter more experienced peers.

The main contributions of this study include the quantification of the contributor's willingness and her environment, in particular, measures of her activities and the amount of effort she is willing to contribute, and measures of micro- and macro-climate. The model quantifying the relationships among attitude, environment, and the odds of becoming a long-term contributor has a potential to significantly improve software practice. For example, projects might adjust their practices to attract and retain contributors by, for example, devoting more attention to new contributors who start by reporting an issue. On the other hand, the newcomers who want to be welcomed by the community would benefit by focusing on the quality of their issue reports and on the community-oriented initial interactions. 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 http://www.passion-lab.org/projects/developerfluency.html.

In Section II we review related work. The project context and the methodology are described in Section III and Section IV presents our findings. We consider the limitations in Section V and conclude in Section VI.

II. Related work

Enormous effort over past decades was spent in attempts to unravel the possible relationships between job performance and its hypothesized antecedents in management science, cognitive science, and psychology. For example, it is widely believed that the characteristics responsible for exceptional performance are innate and are genetically transmitted [5]. However, Ericsson et al [6] showed that many characteristics once believed to reflect innate talent are actually the result of intense practice extended for a minimum of 10 years. Also, environment, for example, access to teachers, training material, and training facilities, rather than talent, are found to be the important factors determining the initial onset of training and ultimate performance [6]. Since Hawthorne studies [7], researchers have examined the effect on performance of formal and informal groupings, peer pressure, roles, norms, cohesiveness, goals, rewards, feedback, task characteristics, and other variables.

The most obvious conclusion from these varied approaches is that the variables known to influence individual task performance are numerous and varied. To summarize the existing theories, Blumberg and Pringle [3] proposed a three-dimensional-interaction model of work performance, in which capacity, willingness, and opportunity are recognized as three interacting dimensions that account for the performance at the individual level. Capacity refers to the physiological and cognitive capabilities that enable an individual to perform a task effectively, including ability, skills, level of education, and endurance. The psychological and emotional characteristics that influence the degree to which an individual is inclined to perform a task comprise the willingness dimension including motivation, personality, and attitude. Opportunity consists of the particular configuration of the field of forces surrounding a person and her task that enables or constrains that person's task performance and that are beyond the person's direct control.

In this study we borrow this three-dimensional framework for one reason – ability, motivation/attitude and environment are the most popular concepts used by scholars in software engineering who have been at work attempting to predict developers' performance. For example, Curtis [8] claimed that the individual differences among project personnel account for the largest source of variation in project performance. Couger and Zawacki [9] identified how differences in the motivational structure of programmers interacted with the kinds of jobs they were assigned. They found that programmers had higher needs for personal growth and personal development than those in any other job category measured. Environmental factors affecting developer's performance include organization variability [10], coworkers [11], and communication media [12]. Studies of newcomer experiences by Dagenias et al. [13] identified early experimentation, internalizing cultures, and progress validation as three primary factors facing developers joining new projects. Earlier we have discovered [4] that the project's relative sociality when a developer joins impacts the probability she will become a long term contributor in that project.

In summary, ability, attitude, and environment comprise the three dimensions that, in literature, are often suggested to account for the developer's performance. However, the quantification of these relationships in software development contexts has been elusive.

III. Methodology

In this study we perform qualitative and quantitative analysis of two OSS ecosystems, Gnome and Mozilla[6]. A qualitative investigation is used to understand joiner behavior and to help design suitable measures of attitude and environment. Gnome and Mozilla have been extensively studied

[5]Logistic regression is a common way to model proportions where the proportion is related to predictors via a logistic function.

[6]Both contain a number of sub-projects

Table I
PROJECTS

Project	Years	MLOC [7]	Domain	Cntrbtrs
Gnome	10	7.9	UI	156,332
Evolution		.8	Calendar&Mailbox	21,041
Nautilus		.1	File manager	17,430
Epiphany		.1	Browser	3,716
Mozilla	12	20	UI	187,333
Firefox		5.3	Browser	47,690
Thunderbird		1.1	Mailbox	12,993
Calendar		.8	Calendar	4,130

in the past. We, therefore, feel an obligation not to interfere with the work of project participants, to the extent we can rely on the results obtained in prior studies. Consequently, we chose to base our qualitative study primarily on recorded artifacts in the issue tracking system, project web pages, the existing literature, and the relevant websites. For questions where we could not obtain needed information from existing sources, we conducted a small survey. Most of information in open source projects is public and we, therefore, feel that such primarily record-based qualitative investigation would be able to capture the essential features of the investigated phenomena.

Our quantitative study analyzes issue workflows in order to establish the relationship between contributors' probability of becoming LTCs and the factors that measure their capacity, willingness, and environment.

We start from describing the project context in Section III-A, introduce the qualitative study in Section III-B, and the workflow analysis in Section III-C.

A. Context

Gnome and Mozilla implement user interface functionality, and have more than 10 years of history, as described in Table I. Some major sub-projects in each ecosystem are also shown in the table. Evolution is the largest Gnome project, and Firefox is the largest of Mozilla's project. Note, that both ecosystems have a browser and a mail client.

B. Qualitative Study

For reasons noted above, we chose to do our qualitative study primarily based on digital records via the following procedure:

- We read the existing literature, particularly about Gnome and Mozilla, e.g., [14], [15], [16], to understand the project context and practices;
- We inspected the project web site looking for the project-related information, for example, the standard workflow of resolving issues. We also looked at the sub project web-pages, searched for relevant information, e.g., the practices used to report and resolve an issue;
- We sampled 40 people (20 non-LTCs and 20 LTCs) from each ecosystem, and carefully read the defects they were involved in, particularly at the time of joining, to understand the joining process and joiner experiences.

We also wanted to get a broader understanding of how people think of the factors influencing individual performance in software projects. We, therefore, communicated with people (we know) from different companies including Google, Microsoft, Tengxun (a big Internet company in China), and Kingrain (a small software company with 13 developers in China). We asked the following question:

- What is the factor that you think has the most influence on the individual's performance?

After we obtained the answer, we asked the following question eliciting response about the factors not mentioned in the first answer:

- Do you think environment/ability/willingness affect individual's performance as well?

Finally, we still felt the need for an independent reality check and sent a small survey to a few participants. We randomly selected eight people from each project and sent them emails containing issues they reported and the following questions:

- What motivated you to report/comment on these issues? Did you report because of your personal interest or because of the business requirements you served at that time?
- Are these the only issues you have experienced? If not, why did you report only these ones?

We obtained one reply from each project, (five emails couldn't be delivered) giving us the response rate of two out of the eleven delivered surveys or 18%.

C. Issue Workflow Analysis

We obtained issue tracking data of Gnome and Mozilla. Traditional software projects use an Issue/MR (Modification Request) system to track defects, enhancements, and other project tasks [8]. The primary users are developers and testers. Customer issues are typically tracked in a separate system and only a very small subset that requires code changes, may be copied/imported into the system used by software developers. In contrast, in OSS software projects the issue tracking systems not only track tasks for developers and testers, but also track issues raised by end users and by down-stream projects. Each issue/MR has a history, from the time somebody reported it until the time somebody closed it (it also may remain open at the time of the study). During that period a sequence of events takes place: MR is created, assigned, resolved, tested, and closed. It may also be reassigned, its attributes changed, comments, debugging

[7]data from ohloh.net

[8]http://en.wikipedia.org/wiki/Issue_tracking_system

traces, etc added. Each such event has an associate date, time, the type of action, and the email/name of the developer performing the action. We consider each such transition as an artifact-mediated communication between two of the adjacent performers.

Crawlers were written to obtain the MR histories and details from their web pages. Both Gnome and Mozilla use Bugzilla to track issues. We obtained information for all issues in XML format as well as the activity history for each project from all the sources 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 158,244 user ids and 517,801 MRs were in Gnome, and 200,655 user ids and 620,511 MRs were in Mozilla. These MR traces record people's actions related to issue resolution, and therefore, are likely to reflect people's characters. The climate of a project is determined by participants' behavior, therefore, the same traces are also likely to reflect the climate of the project at any specific time. We investigate the workflow of people resolving issues to measure people's capacity, willingness and opportunity.

In general, people follow a protocol to resolve issues, e.g., Gnome defines the standard steps of triaging on its website [9]. According to it, an issue is reported (born) in an UNCONFIRMED state, until it is confirmed, and its state is changed to NEW (or, it may be immediately resolved and the state is changed to RESOLVED). When additional information from the issue reporter is needed to proceed further in fixing this issue, the state would be changed to NEEDINFO. Later this issue would be added to an assignee and needs be resolved. The transition from one state to another may involve a change of actors. Newcomers take different actions because of their capacity and willingness, as described in later sections. Notice, not everybody is allowed to modify the state of an issue. We have verified that a new contributor is allowed to modify the state only for the issues she reports, but she is able to comment on any issue. Mozilla's triage process is similar [10]. However, in practice the process is a bit different. Here we focus on determining the roles people play in the issue resolution process.

Each "RESOLVED" issue has a resolution, e.g., FIXED, DUPLICATE, INCOMPLETE, INVALID, or WONTFIX. The resolution types vary between Gnome and Mozilla, e.g., Mozilla resolution EXPIRED is used in similar cases as OBSOLETE in Gnome. As the name suggests, FIXED means the bug is fixed, DUPILICATE means the reported bug is a duplicate of some other bug, INCOMPLETE means the reported information is not sufficient to reproduce the bug, INVALID means this is a invalid report, and WONTFIX means the reported issue is not relevant enough to be fixed.

[9]http://live.gnome.org/Bugsquad/TriageGuide

[10]https://bugzilla.mozilla.org/page.cgi?id=fields.html

Issues with resolution FIXED represent 32% of Gnome's and 36% of Mozilla's resolved issues. DUPLICATE represent 36% and 24%, and INCOMPLETE 15% and 6% of all resolved issues.

IV. Results

Contributors interact with their environment when they join the project. That interaction is mediated by their capacity and willingness and it affects the odds of them becoming LTCs. We start from quantifying contributor's capacity and willingness in Section IV-A, continue with measuring contributors' environment in Section IV-B, and conclude with fitting a logistic regression model of the probability that a newcomer will become an LTC in IV-C.

A. Measuring Capacity and Willingness

Software development is a knowledge intensive activity [17], and the almost universal assumption of personnel managers is that personality has a marked effect on the performance of employees [3]. As an employee from Google commented in our interviews, *"basically, personality determines everything"*. And *"environment is similar for everyone, attitude is a part of personality"*. How about individual ability? *"We are Google, never lack talented guys"*. However, another interviewee (a manager from Kingrain) said *"ability accounts for everything"*, because his best employee is substantially more competent than the remaining employees. As Blumberg and Pringle [3] clarified, personality, attitude, and motivation are variables of "willingness to perform", while ability, experience, and intelligence are variables of "capacity to perform". In this section we measure the capacity and willingness of newcomers to the OSS project, and try to establish if the contributor's willingness measured at the time they join the project varies between participants who will and who will not become LTCs.

Only 2.12% of Gnome and 0.90% of Mozilla joiners become LTCs. And in both projects, more than 70% of contributors are one-time-contributors (OTCs, 70% in Mozilla, 78% in Gnome). These contributors have only a single interaction (e.g., reporting an issue or commenting on an issue) recorded in the issue tracking system. This high level of peripheral participation is consistent with the finding that more than three-quarters of the nearly 13 thousand contributors Lerner and Tirole [18] considered, made only one contribution. We sampled 20 OTC and 20 LTC participants from each project. For ethical reasons, we omitted the participants' names and represent them using numeric aliases, with 1-20 representing OTCs and 21-40 representing LTCs.

We manually inspected all the issues reported by OTCs, and observed two types of behavior. Both of these behaviors were exhibited by contributors who appear to be end users. Some were eager contributors who worked thoroughly and tried to help. For example, gnome-7 reported: *"I've been trying to use libxml++ ... However I found that there are two rather large memory leaks"*. He even committed some attachments to help fix, *"Please let me know if these issues will be addressed"*. Some people proposed specific requirements. For example, mozilla-15 asked to implement a new feature, *"I'm writing to request support for JIF be added to mozilla*". For this he was willing to help – *"I can provide some help for this, but I would like to know if the mozilla developers are receptive to this"*. Another group appeared to be not devoting as much attention to the issues they reported. For example, gnome-12 reported two issues according to the required format but without any extra comments. Furthermore, he/she didn't respond to request for additional information: *"Could you please help fixing this by installing some debugging packages..."* (the issue was resolved with the state INCOMPLETE). These are some of the reasons to believe that these people reported issues (and committed fixes) because they were end users and they were trying to accomplish something relevant with the OSS product they were using. At the same time, they might have found the issues by accident (because they were users), and they were interested enough to report it but were not motivated enough or had no opportunity to contribute more.

For the sampled LTC group, the average number of MRs per person was over 100 (over the considered period, see Section III-C). We, therefore, examined only a subset of their issues, in particular, issues in their first month after joining. Notably, they all were more active as compared to the OTC group: made more comments, and often spent more effort on their issue reports, for example, by including a patch. For example, mozilla-25 reported his first issue on Oct 24, 2007. Based on the comment on that issue: *"Reporter, could you reproduce on FF3 RC?"*, we assume that he was not known by the community at that time. Less than one month later, on Nov 21, mozilla-25 created attachments for a couple of issues. Apparently, he already won trust from the existing developers by, perhaps, submitting good quality patches, because a developer applied to obtain an SVN account for mozilla-25 the same day (through Bug x: Create localizer LDAP/SVN account for mozilla-25). A similar example in Gnome is gnome-30, who reported his first issue on Mar 20, 2004. This report was committed with a couple of extra attachments showing the details needed to understand and reproduce the issue, and the issue resolution was "FIXED". Another example is gnome-33, whose first report was *"Patch to get access attributes for nested class/struct/union"*. In other words, the first action he did is to commit a patch in the form of an issue report, and it appears to have been useful, because the responsible developer responded – *"I'll include it in the first CVS release"*. It is worth noting that four people from the LTC sample ($4/40 = 10\%$) appear to have been in the development team from the very beginning. For example, the first participation of mozilla-22 is a comment: *"Checked in code for spec-compliant implementation of webclient on Solaris ..."*

In summary, all OTCs and most LTCs appear to be product users at the time of joining. The two survey responses were *"I use a lot of Open Source software both at work and at home (including Mozilla)"*, and *"Around that time, I did the final move away from Windows and replaced Windows on my desktop PC at home (with Gnome)"*. This observation is consistent with the findings in project Fedora: *"74% said they were first users of Fedora and then became contributors to the project"*[11]. Moreover, both groups show ability and willingness to contribute — out of millions of Mozilla/Gnome users, only a very small proportion is going to contribute, no matter if the contributions are "in the form of bug reports, suggestions or occasionally code contributions". Also, quoting from our survey: *"I did report the issues because of my personal interest"*, *"as a technically knowledgeable user I feel a responsibility to give back when I am able"*.

However, what are the differences among contributors, in particular, between OTCs and LTCs? The evidence appears to favor willingness, i.e., the extent to which people are getting involved in helping the project. First we consider the number of tasks, e.g., number of comments she makes. These comments might be an interpretation for a confusing report, or suggesting a possible solution, or explaining the benefits of a proposed new feature. We assume that the number of tasks primarily reflects the skills the participant has accumulated prior to joining the project, i.e., her capacity, but also, partly, how willing she is to contribute.

There are two other aspects representing the extent of participation — what type of tasks a participant takes on and how much effort she provides. The activities a participant undertakes, e.g., attaching the screenshot when reporting an issue is a sign that she has a community-friendly attitude. Starting from a comment instead of reporting an issue also reflects such an attitude — it means she is intentionally getting involved, perhaps by first finding a similar issue and commenting on it instead of simply reporting an issue she encounters as a user. On the contrary, reporting an issue through a crash-reporting tool such as Bug-Buddy is extremely easy and does not demonstrate a great amount of desire to get involved. When an application using the GNOME libraries crashes, Bug-Buddy generates a stack trace and invites the user to submit the report. This requires little effort from the user, simply filling a few fields and clicking "Submit" button. In contrast, alternative 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% of the joiners, who had their first issue reported via Bug-Buddy, eventually became

[11]http://www.cyber-anthro.com/beta-an-exploration-of-fedora%E2%80%99s-online-open-source-development-community/

LTCs, while more than 4% of the joiners who started with a Bugzilla report became LTCs.

Even though contributions appear to be universally valued in open source projects, the following statistics casts doubt on that assumption, or, at least, suggests that some contributions provide much more value than others. Approximately 90% of the OTC reports ended up without any change (i.e., without resolution FIXED, 91% in Mozilla, and 90% in Gnome), of which a big proportion were duplicates (47% in Mozilla and 35% in Gnome). However, for the defects reported by LTCs, a high proportion were FIXED (51% in Mozilla, 52% in Gnome), and approximately 15% in both projects were duplicates.

This analysis shows how the issue resolution types can be used to measure the value a contributor provides and to reveal her attitude. Specifically, fixed issues help improve the product quality and might enhance community morale by showing some accomplishment. Meanwhile, the remaining issues might waste the limited time of the few core developers.

On the other hand, when a contributor encounters an issue, if she spends more time to search for similar issues and the relevant resolutions, the more likely she is to find exactly the same or similar issue. There is less chance, therefore, that the issue will end up with DUPLICATE resolution. The same argument applies for issues with resolution INCOMPLETE — responding and providing enough information for others to reproduce the issue would make it more likely that the issue will be fixed. In other words, if a person puts enough effort, the issues she reports have more chances to be fixed and, thus, improve the quality of the product. Consequently, the fraction of reported issues that are fixed (in the first month from joining) is an indication of effort a contributor provides. Therefore:

Observation 1: A contributor's willingness (and capacity), can be measured by the number and types of tasks (e.g., reporting or commenting on an issue) she participates in and by the effort she provides to resolve these tasks.

B. Measuring Environment

Early literature on job performance predictions did not consider the environment: "since performance is ultimately an individual phenomenon, environmental variables influence performance primarily through their effect on the individual determinants of performance – ability and/or motivation" [19].

However, available evidence indicates that certain environmental factors beyond the individual's control play a far stronger role influencing her performance than was generally acknowledged then [3]. The more important of these involve what is known in normative decision theory as states of nature and actions of others, and suggest a clear recognition that, in addition to social, psychological, and physiological determinants, behavior (performance) also depends on the help or hindrance of uncontrollable events and actors in one's environment.

In our case we separate two aspects of the environment: macro-climate and micro-climate to account for states of nature and actions of others. Macro-climate represents the overall project environment that is the same for everybody in the project. Micro-climate represents the conditions that each individual encounters and varies among participants.

We start from measuring macro-climate. Based on the literature and our experiences, product popularity, project task density and project sociality are important elements. The market value of a product comes from its usage. In other words, the number of users ($users$) reflects the product's market proceeds, that will affect the funding (tools, equipment, materials, supplies, and pay) for the project and the degree of interest people would devote to it. Consequently, it will likely affect the contributors' stay with the project. We measure project popularity via its user population each month. For example, Firefox is the primary product in Mozilla and we use its user base as proxy for Mozilla user base. We obtained the user population by multiplying the market share of Firefox[12] by the total number of Internet users[13]. 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 the 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., number of participants each month ($newJoiner$), or relative sociality (RS) [4]. Figure 1 shows the evolution of macro-climate components in Mozilla. We were able to obtain Internet user estimates starting from Dec, 2000. Figure 1 also marks the calendar times of important releases of Firefox, e.g., in Jan, 2010, Firefox 3.6 was released. We have divided the user numbers for each month by the highest number of users over the entire period. Thus, this variable ranges in value between zero and one. Notice, in order to present the curves on a similar scale, we normalized each, e.g., by dividing $newJoiner$ by 50, as shown in the legend of the chart. The project's popularity grows from the start of the project, because Internet user population grew and Firefox share has increased. The task density is high when the project is close to its release date, and participant density is high when the task density is high.

These components of macro-climate are the same for all project participants at a particular time. Person's micro-climate may be characterized with the following Chinese proverb: "He who stays near vermilion gets stained red, and he who stays near ink gets stained black". We, therefore, consider people in her workflow network, i.e., people she

[12]http://en.wikipedia.org/wiki/Usage_share_of_web_browsers
[13]http://www.internetworldstats.com/stats.htm

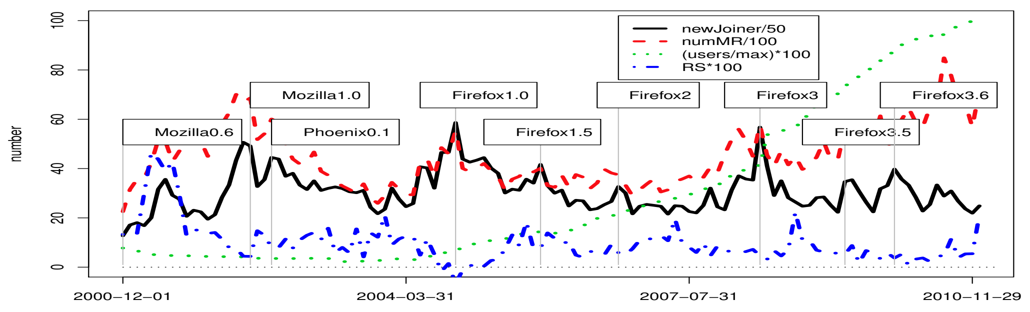

Figure 1. Macro-climate in Mozilla

interacts with, to create a unique environment for her. In other words, the characters or actions of these people, and her relationship with them, constitute the micro-climate for this person. We chose the initial size of a participant's social group, their productivity, their social clustering, and the attention they give 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 meet more people. 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. According to [20], we assume that if a person has more peers, she is more likely to attach to the project, therefore she is more likely to become an LTC. On the other hand, the performance of her peers is likely to affect her performance, as in an old saying: "a cat gets stronger with a tiger." For example, Mockus [21] found that more productive mentors lead to more productive followers.

We consider the social clustering to be the amount of overlap among the workflow networks of peers. For example, Contributor Alice has two peers, Dragon and Tiger, and Dragon meets Lion and Bear, Tiger meets Lion and Deer, 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 counted twice), therefore her social clustering is $\frac{6-4}{4}$. The underlying assumption is that if a person's peers have more in common (share more colleagues), it is more likely that they have similar project experiences and 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 for a newcomer to learn and to become more effective. It might also increase her work satisfaction and the willingness to stay. For example, Mayer [20] found the willingness to trust others was significantly related to the behavior and performance of individuals.

Humans need attention from other people, and developers are no exception, notwithstanding common stereotypes. Perhaps, the more attention a newcomer could obtain from the existing project members, the more likely she will stay with the project for a long time. We measure the amount of attention through the duration of time between the newcomer's first action until somebody responds. The response delay that is too short may not bode well in terms of attention. A very short 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.

Therefore:

Observation 2: A contributor's environments may be measured via macro- and micro-climate. Macro-climate represents environment shared among participants, and consists of project popularity, project task density and project sociality. Micro-climate represents environment unique for each person and consists of the initial size of peer group, their productivity, their social clustering, and the attention they devote to this new participant.

C. The Chances of an Individual's Success

We investigate the influence of the ability, willingness, and environment on the chances of an individual's success in the project by fitting a logistic regression model specified in Equation 1 with 125,665 observations in Gnome and 130,471 observations in Mozilla. The response is the indicator of a new participant becoming an LTC and the predictors include measures of her ability, willingness, and environment described above (Notice not all attributes are presented because of the correlation, e.g., project task density is heavily correlated with RS). Each observation represents one project participant, with the predictors calculated over her first month from joining shown in Table II. 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 she encounters in her first month's workflow. 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 is not correlated with other predictors in both projects.

Popularity ($nUsr$) represents the number of Firefox users in Mozilla, as described in Section IV-B. We did the following to approximate the historic numbers of Gnome users. First we obtained the estimates of the fraction of linux users[14] [15]. Then, we used surveys of desktop choices for the period between 2003 and 2008[16]. For the 2009 to 2011 we approximated Gnome users by the fraction of Ubuntu users[15]. Eventually we multiplied the market share of Ubuntu/Gnome by the estimates of Internet users[13] to approximate Gnome user numbers.

We used $GotFix$ to represent the value contributor provided to the community. Having at least one of the reports to be fixed indicates a tangible improvement of product quality.

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 Bug-Buddy) 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 for high willingness to contribute.

We used $LckAttn$ to represent an extreme situation of a too rapid response (within one hour).

Predictor prj is a sub-project indicator (not shown in the table) of the ecosystem the participant starts with, e.g., Evolution of Gnome, Firefox of Mozilla. It explains $1-2\%$ of the deviance and was added to account for the variation among sub-project environments.

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

Tables III and IV contain fitted values for Gnome and Mozilla. 23% of the deviance is explained in Gnome model and 19% in Mozilla. The second column has the estimated coefficients, and the third standard errors. All predictors are significant (at < 0.01 level), except for RS in Mozilla. The forth to sixth columns show practical importance of the predictor in determining the LTC probability through effect sizes. In logistic regression the effect size is odds ratio (i.e., the sixth column) for the mean value of the predictor (i.e., column labeled by x) and for the mean value plus the standard deviation (i.e., column labeled by x_{alt}). That's the case for predictors $nUsr$, RS, $pShared$, and $PeerPerf$.

[14]http://en.wikipedia.org/wiki/Usage_share_of_operating_systems
[15]http://stats.wikimedia.org/archive/squid_reports/
[16]http://www.desktoplinux.com/news/NS8454912761.html, http://www.desktoplinux.com/articles/AT2127420238.html

Table II
PREDICTORS FOR PARTICIPANT i

Dimension	Predictor	Description
Willingness (and capacity)	nCmt	Logarithm of the number of comments +1
	GotFix	At least one of the issues reported by i was fixed.
	withBB FNotRep	First report by i uses a crash reporting tool 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 the size of i's peers' groups $\ln \| \cup_{p \in Peers(i)} Peers(p) \| + 1$ where $Peers(p)$ is the peer group for p
	pShared	Logarithm of the social clustering of i's peer group $\ln \frac{\|\cup_{p \in Peers(i)} Peers(p)\|}{\sum_{p \in Peers(i)} \|Peers(j)\|} + 1$
	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

Table III
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

For predictors with low discrete 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. Two hypothetical Mozilla contributors: Alice with one comment in her first month and Bob with four comments (90th percentile of $nCmt$ is $\ln(4+1)$ or four comments), therefore the odds of Bob becoming an LTC are 112% higher than odds for Alice if their remaining predictors have values specified in the forth column of the table.

In summary, the probability of becoming an LTC is associated with the contributor's willingness and environment. Specifically, starting from comments instead of reports, reporting with Bugzilla instead of Bug-Buddy, or reporting any sensible issue that gets fixed double the odds of becoming an LTC. Regarding micro-climate, low attention in the form of too rapid response would reduce the odds by 28% in Mozilla and by 39% 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

Table IV
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

the odds by 95% and 20%. Increasing the social clustering by 0.11 in Gnome and by 0.2 in Mozilla leads to 22% and 48% increase in the odds. For the macro-climate, product popularity is associated with lower odds that a contributor becomes an LTC — increasing the number of users by 34% in Gnome and by 26% in Mozilla reduces odds by 49% and by 15% respectively. Project's RS is associated with lower odds of becoming an LTC in Gnome (0.326 of RS increase leads to 27% decrease in the odds) but higher odds in Mozilla (0.0992 of RS increase leads to 7% increase in the odds). Also, having the size of the peer group increasing from the median to third quartile is associated with a small decrease in the odds (6%) in Gnome and an increase (14%) in Mozilla. In summary:

Observation 3: The probability of a newcomer becoming an LTC is associated with her willingness and environment. Her pro-community attitude that determines her choice to start with a comment instead of a report or to report with Bugzilla instead of automatic tool, and the amount of effort she provides to the community, are associated by 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.

As was shown, contributor's attitude (e.g., starting from a comment instead of a report, or getting at least one reported issue fixed) is much more strongly associated with her chances of becoming an LTC as compared to other factors. It's consistent with opinions of Google and Microsoft employees we interviewed. According to them, what matters is attitude, because either environment or ability do not pose obstacles for individual improvement. However, for Tengxun employee, what matters the most is experience, because less motivated people will not stay employed. For a small company like Kingrein, the most important aspect appears to be ability, because it is the most obvious differentiator among the employees there (though the manager also emphasized that the attitude makes a big difference as well). In other words, when people are in the same environment, what matters is person's capacity and/or willingness. If people are compared in different environments, the environment might be the primary determinant: Kingrain manager agreed that if his employees worked at Google, they might perform better because they had chance to work with more talented people. Consequently, we have:

Observation 4: The dominant factor in capacity, willingness and environment that affects the probability of a newcomer becoming an LTC depends on the project context.

V. LIMITATIONS

The first limitation relates to the consistency and accessibility of the data in the Bugzilla of Gnome and Mozilla.

First, the web crawling appears to miss a substantial fraction of defects. We, therefore, obtained the entire range of defects from 1 to 645899 for Mozilla and 639379 for Gnome. Some defects were either not public or not obtainable: 121578 in Gnome and 25388 in Mozilla. We also compared different snapshots extracted at different times, and verified to make sure that the later snapshots include the issues in the earlier ones. While there were no problems obtaining Mozilla data, Gnome Bugzilla prevents retrieval of large numbers of issues with complete email address of contributors. We, therefore, had to rely on public extracts of Gnome Bugzilla data to map logins to individuals.

Second, we assume that a login (for Gnome) and an email (for Mozilla) is a unique representation of a single person. However, multiple people may share the same login or a full email, and a single person may use several email addresses. To deal with this issue, we used two approaches. First, we identified "generic" logins, e.g., "mozilla", "gnome", "bugzilla*", "*maint" because we focused on ordinary contributors. Second, we used the full name of the participant associated with each login/email to identify all logins that had two or more names associated with them (multi-name logins). We also identified full names that had more than one login/email associated with them (multi-login names). To verify that our conclusions are not affected by this assumption, we fit our models on three datasets: the original data, the data without generic logins, and the data without generic and multi-name logins and with all logins having multi-login names replaced by the corresponding name. The results were similar and we reported results using the original data.

Third, the data itself 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. 995 MRs in GNOME and 601 MRs in Mozilla had intermediates states missing, but excluding them didn't have any noticeable impact on the results.

The second limitation relates to the measures we construct: do they reflect the intended concepts and can we measure each concept separately from other concepts? Individuals and their environment are notoriously difficult to measure because of the variability among individuals and the ambiguity of concepts such as willingness and opportunity. The issue tracking systems, however, provide a practical opportunity to observe the activities individuals engage in and infer the effort they spend to conduct them. From these basic measures it is possible to estimate their capacity and willingness. Second, it is almost impossible to measure one separate dimension. People's activities we observe are always the combined effect of multiple dimensions. For example, number of issues a participant is involved in shows not only her experience, but also her willingness to contribute, and maybe how buggy that product release is at that time. Fortunately, as Ericsson et al [6] claimed, 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 deliberate practice is through willingness. Therefore we encompass capacity and willingness into a single dimension — we refer to it as willingness — after all, willingness determines how much an individual would get involved in a volunteer activity.

From the internal validity perspective we checked the assumptions for the logistic regression and log-transformed the predictors. While only $19 - 23\%$ of the deviance is explained, this is, in fact, a very good fit given that only $0.9\% - 3.6\%$ of the participants become LTC (in Mozilla-Gnome). From the prediction perspective, if we randomly picked 1% of participants and compared with model predictions, our precision would improve 25-11 times. For example, in Mozilla, among developers with the top 1% predicted probability 22.15% are LTCs, while for a random predictor the precision would be 0.9%.

From the external validity perspective, the way Gnome and Mozilla are operating is not unusual for open source projects. The models for both projects are quite similar. 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.

Its 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.

VI. Conclusions

In this study, we have tried to address the following research question: what impacts the chances that a new joiner to a software project will become an LTC? Is it related to differences of persons' characters, variation in project climate, or, to the interaction between individuals and their environments? We measured the behavior of individual participants in Gnome and Mozilla using issue workflow, and modeled how the differences in their behavior affected the probability of the participant becoming a valuable contributor to the community.

We found that the main differences among participants were in their capacity, willingness and opportunity to contribute at the time of joining. A participant's capacity and willingness is measured through the activities she takes on, i.e., the number and type of tasks she starts with, and the effort she puts into her contributions, e.g., the fraction of reported issues that were ultimately fixed. The opportunity is measured as environment's macro-climate that was shared among all participants and micro-climate that was unique for a person. In particular, product popularity, project task density, and project sociality constitute measures of macro-climate, while the initial size of peer group, their performance, their social clustering, and their attention to the new participant constitute micro-climate. We found that the probability of a newcomer becoming an LTC to be associated with person's willingness and environment. Most importantly, her pro-community attitude represented by her first contribution being a comment on an existing issue instead of a bug report or a report through Bugzilla interface instead of a crash-reporting tool double her odds of becoming an LTC. Having any of issues reported during the first month to be fixed has the same effect. Her micro-climate represented by low attention of too rapid response, and her macro-climate represented by the increased project popularity reduce her odds. Project's relative sociality and individual's peer network size had opposite effects in the two projects. This may reflect some inherent differences between practices of Gnome and Mozilla that need further study.

These finding may help individual participants to understand what their own roles are and find the best ways to contribute. It is also likely to help OSS communities to adopt better strategies to attract and retain newcomers. Specifically, the probability of staying longer is associated with how much value a new participant provides to the project by commenting, putting more effort into issue reports, and by the amount of attention the project provides to the newcomer. Ironically, it is during times when projects are popular, thus overwhelming the mentors, the community needs to put extra effort to retain newcomers.

Acknowledgment

The work was partially supported by National Basic Research Program of China Grant 2009CB320703, the National Natural Science Foundation of China Grant 61121063 and the Nature Science Foundation of China Grant 61073016.

REFERENCES

[1] M. Zhou and A. Mockus, "Developer fluency: Achieving true mastery in software projects," in *ACM SIGSOFT / FSE*, Santa Fe, New Mexico, November 7–11 2010, pp. 137–146.

[2] D. H. Lindsley, D. J. Brass, and J. B. Thomas, "Efficacy-performance spirals: A multilevel perspective," *Academy of Management Review*, vol. 20, no. 3, pp. 645–678, Jul. 1995.

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

[4] M. Zhou and A. Mockus, "Does the initial environment impact the future of developers?" in *ICSE 2011*, Honolulu, Havaii, May 21–28 2011, pp. 271–280.

[5] U. Neisser, G. Boodoo, J. Bouchard, Thomas J., A. W. Boykin, N. Brody, S. J. Ceci, D. F. Halpern, J. C. Loehlin, and R. Perloff, "Intelligence: Knowns and unknowns," *American Psychologist*, vol. 51, no. 2, pp. 77–101, Feb. 1996.

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

[7] H. M. Parsons, "What happened at hawthorne?" *Science*, vol. 8, no. 4, pp. 922–932, 8 March 1974.

[8] B. Curtis, "Fifteen years of psychology in software engineering: Individual differences & cognitive science," in *ICSE'84*, 1984, pp. 97–106.

[9] J. D. Couger and R. A. Zawacki, *Motivating and Managing Computer Personnel*. New York, NY, USA: John Wiley & Sons, Inc., 1980.

[10] A. Mockus, "Organizational volatility and its effects on software defects," in *ACM SIGSOFT / FSE*, Santa Fe, New Mexico, November 7–11 2010, pp. 117–126.

[11] A. Ko, R. DeLine, and G. Venolia, "Information needs in collocated software development teams," in *ICSE 2007*. ACM Press, May 20–26 2007, pp. 344–353.

[12] M. Cataldo, P. Wagstrom, J. Herbsleb, and K. Carley, "Identification of coordination requirements: Implications for the design of collaboration and awareness tools." in *Conference on Computer Supported Cooperative Work CSCW'06*, Banff, Alberta, Canada, 2006.

[13] B. Dagenais, H. Ossher, R. K. E. Bellamy, M. P. Robillard, and J. P. de Vrie, "Moving into a new software project landscape," in *ICSE 2010*, Cape Town, South Africa, May 1-8 2010, pp. 275–284.

[14] A. Mockus, R. T. Fielding, and J. Herbsleb, "Two case studies of open source software development: Apache and Mozilla," *ACM Transactions on Software Engineering and Methodology*, vol. 11, no. 3, pp. 1–38, July 2002.

[15] D. M. German, "The gnome project: a case study of open source, global software development," *Software Process: Improvement and Practice*, vol. 8, no. 4, pp. 201–215, 2003.

[16] P. A. Wagstrom, "Vertical interaction in open software engineering communities," *Phd thesis, Carnegie Mellon University*, vol. CMU-ISR-09-103, March 2009.

[17] P. Robillard, "The role of knowledge in software development," *Communications of the ACM*, vol. 42, no. 1, pp. 87–92, 1999.

[18] J. Lerner and J. Tirole, "Some simple economics of open source," *The Journal of Industrial Economics*, vol. 50, no. 2, pp. 197–234, 2002.

[19] L. L. Cummings and D. P. Schwab, "Performance in organizations: Determinants and appraisal." *Administrative Science Quarterly*, vol. 18, no. 3, pp. pp. 412–414, 1973.

[20] R. C. Mayer, J. H. Davis, and F. D. Schoorman, "An integrative model of organizational trust," *The Academy of Management Review*, vol. 20, no. 3, pp. pp. 709–734, 1995.

[21] A. Mockus, "Succession: Measuring transfer of code and developer productivity," in *2009 International Conference on Software Engineering*. Vancouver, CA: ACM Press, May 12–22 2009.