Abstract

Open source development is labelled with free source, fast evolution and extensive user collaboration. Previous studies primarily touched on user activities in a few well-known open source projects but lack the empirical data to represent important issues facing the open source development community. This paper reports on a survey-based study that investigated user collaboration in open source development. Among the many interesting results, we found open source users themselves to be very experienced software developers. They are highly motivated to make their own contributions, work closely with developers on various tasks, and use electronic communication tools extensively. Users with different development experience and contributing to different category of projects tend to exhibit different motivations and behaviours. These findings confirm the differences between traditional and open source user collaboration and their implications for future open source evolution and management are given and discussed in this paper.

Keywords: collaborative software development, open source, user collaboration, survey

Introduction

With the successful delivery of products such as Linux, Apache and Sendmail, the open source development model has been attracting more and more interests from both practitioners and researchers. Open source is a simple means of releasing software with source code, but one that brings a series of new social and technical challenges including licensing, distributed development and project management, commercial adoption and extensive user collaboration. Various research methods have been adopted to investigate the phenomena of open source development, including general descriptive discussion (Raymond 1999; Hars and Ou 2001; Cubranic and Booth 1999; Augustin et al. 2002), case studies (Mockus et al. 2002; Lakhani and Hippel 2003), and surveys (Zhao and Elbaum 2003; Lakhani and Hippel 2003; Hertel et al. 2003).

With the classic statement ‘Given enough eyeballs, all bugs are shallow’, (Raymond 1999) clearly indicates the essence of open source development — user involvement. The concepts of user involvement and innovation are not new to open source. (Hippel 2001) made an excellent comparison of user communities in open source software and traditional manufacturing. Two conditions, incentive to innovate and incentive to voluntarily reveal innovations, are said to be common for both communities favoring user innovation. The difference is that generally the diffusion of open source innovations, unlike with physical products, does not need manufacturer support. Traditional software engineering disciplines already deem user involvement as key to avoiding software project failure (Jiang et al. 2002) and key to success (Clavadetscher 1998). Many methods and collaborative environments were proposed for better communication and collaboration between developers and users (DeBelli and Haapala 1995; Keil and Carmel 1995), though in open source user involvement carries different meanings. Compared to more controlled and limited user participation, open source takes a more spontaneous, less organized, and in-depth form of user involvement. This type of user involvement usually exceeds what users are supposed to do in traditional software development settings and the boundary between users and developers become blurred as opposed to developer — user links in commercial settings (Keil and Carmel 1995). The reason is that users participate deeply in development tasks like finding bugs, suggesting new features, inspecting source code, and submitting patches.

Authors

Luyin Zhao
(luyin.zhao@philips.com) is a software researcher at Philips Research USA and a part-time PhD student at the State University of New Jersey — Rutgers. His research interests include software engineering in open source, healthcare decision support systems, and distributed-object computing technologies.

Fadi P. Deek
(fadi.deek@njit.edu) is Associate Dean of the College of Computing Sciences, Director of the Information Technology Program, and Professor of Information Systems at New Jersey Institute of Technology. He is also a member of the Graduate Faculty — Rutgers University PhD Program in Management.

Dr Deek’s research interests include Learning/ Collaborative Systems, Software Engineering, Programming Environments, and Computer Science Education.
yet taking less responsibilities than core development teams. This user group is also defined as ‘innovating lead users’ in (Hippel 2001; DeFranco-Tommarello and Deck 2002). From this perspective, the word ‘user collaboration’ is more appropriate to reflect this situation.

Collaborative software development is one of the main application domains of groupware and group support systems. (Robillard and Robillard 2000) use a case study to illustrate four types of collaborative tasks in software engineering. (Deck and McHugh 2003; DeFranco-Tommarello and Deck 2002) give a good summarization of groupware technologies for collaborative software development. Core open source development teams are more likely to adopt certain type of coordination mechanism (Wu and Lin 2001) and even groupware. However we have enough reasons to believe that for open source development, particularly the part of development activities that get users involved, it is inappropriate to copy and paste groupware and collaboration mechanisms designed for traditional software development. For example, due to the voluntarily nature and undefined team roles, there is little chance to do synchronous collaborative tasks among members and it is almost impossible to formally split and assign development tasks to individuals. Currently, open source collaborative development is achieved by taking advantage of lean electronic media such as email, version control systems, and web pages (Yamauchi et al. 2000; Augustin et al. 2002) as opposed to rich media for group communication and development (Daft and Lengel 1986; Kuwana et al. 1996). So far, how well the current open source setting facilitates user collaboration remains unknown.

To address this problem, we first need empirical data as to how users perform and behave in this collaborative development environment. Although some research articles have explored user collaboration in certain typical and well-known open source projects (Lakhani and Hippel 2003; Mockus et al. 2002), in general there is a lack of extensive and empirical studies as to how user collaboration is taking place in a wider range of open source projects. To this end, we conducted a survey study to uncover the type of people that actively participate in open source development as users, to learn about what motivates them, the type of tasks they do, how they collaborate with development teams, the effectiveness of such collaboration, and in what direction open source collaborative environments should be improved and what this means to open source projects sponsored by commercial vendors.

The next section introduces some related work of studying open source development using empirical data, followed by the survey methodology description. We then present the study’s major findings. The paper ends with a discussion on the implications of this study to the future evolution of open source development.

### RELATED WORK

Recently, two empirical research methods, case study and survey, have been used by researchers to investigate the phenomena of open source development. In these studies, the puzzle of people’s motivation to contribute to ‘free’ software is one of the primary interests. Another direction is more development-process oriented, with the objectives of finding out unique development process(es) followed by open source projects and how they are different from commercial software development.

Hars and Ou (2001) assume two types of motivation that account for people’s participation in open source projects — internal factors (intrinsic motivation, altruism and community identify) and external factors (future rewards, including selling products, human capital, self-marketing, peer recognition and personal need). The assumption was verified by a questionnaire survey to the open source community with 81 responses. In a similar survey-based study, (Lakhani and Wolf 2003) conclude that enjoyment-based intrinsic motivation, user need, intellectual stimulation and programming skills improvement are among the major motivations.

Another survey-based case study on Apache web server, (Lakhani and Hippel 2003), investigated how ‘mundane but necessary’ tasks are done among open source users and what motivated users doing so. Using Apache Usenet log data and 336 replies from a questionnaire-based survey, the study revealed the pattern of activities in Apache help system. It shows that the user-to-user help system functions effectively, and most notably, 98% of the helping efforts provide direct learning benefits to the help providers themselves.

A recent work (Bonaccorsi and Rossi 2003) shifts attention of motivations from the individual to firm, namely investigating why firms are motivated to invest in open source and base their business models on open source. Significant differences are reported to be found between the motivations of individuals (primarily social motivations) and firms (primarily economical and technological motivations).

In two case studies of development in Apache and Mozilla, Mockus et al. (2002) provide information about basic parameters of the process that these two projects employed and the outcomes of these processes. One hypothesis proved by both projects says that open source development usually have a core of developers who control the code base and the number of this core does not exceed 10 to 15 people and they contribute to about 80% of the total work. Another verified hypothesis stated that in successful open source developments, a group larger by an order of magnitude than the core will repair defects, and a yet larger group will report problems. Both hypotheses support the motivation of our study — investigating the characteristics and behaviours of the ‘larger’ group of people outside of the core development team.

Using a similar method to that used in this paper, Zhao and Elbaum (2003), as a more mature study than
(Zhao and Elbaum 2000), investigate the process and quality assurance activities conducted by core development teams. A questionnaire was administered to open source core developers registered in freshmeat.net and sourceforge.net and 229 responses were used in data analysis. This study found that user participation was extremely high, which is ‘generating up to 20% of the changes for almost 50% of the projects, and discovering 20—40% of the faults in 20% of the projects’.

SURVEY METHODOLOGY

Sampling in this research was done on a single open source hosting website www.sourceforge.net for several reasons. Primarily, this website has become the world’s largest open source hosting site with 64,231 registered projects and 651,088 registered users, as of 14 July 2003. This number is much larger than any other hosting site. Also, another reason is that sourceforge has well-structured project management web pages and easy-to-access user contact methods.

As already indicated, this research is concerned with users who download and use open source software as well as actively participate or contribute to the software development. Henceforth, we refer to this group of active users as ‘user’ or ‘users’.

To ensure receiving responses quickly and to avoid unreachable email addresses, we limited our survey to recently active users. Therefore, the sampling starts with the top 200 active (as ranked by sourceforge) projects as of 10 July 2003. For each project, 12 users (12 was used because it seems to be a reasonably large enough number to include users that were active in the past few months) who have most recently posted at least one message, including submitting bug reports, suggesting new features, or offering patches were selected. In selecting actual users, efforts were made to distribute users evenly in the three activities, say four users for each activity. Moreover, in order to study user behaviours across different software categories, we attempted to distribute the number of projects evenly within seven categories as shown in Table 1, with the number of samples eventually obtained. As a result, 73 projects were finally selected and others were dropped because of various reasons such as insufficient number of users and too many projects in the same category. Because some projects fit in multiple categories, the total number of users sampled is 846. Personalized emails were sent to the sample population and 133 responses were received, yielding a respond rate of 15.7%, or on average 1.82 responses per project. We received insufficient responses for the category ‘Games/entertainment’, so it is left out in the following data analysis.

USER CHARACTERIZATION

Group composition, referring to the membership characteristics or subject demographics of a group, is often the first step in investigating a collaborative environment (Deek and McHugh 2003). Group characteristics are typically closely related to the effectiveness of the collaborative environment. Compared with the traditional sense of software development, with controlled user groups of known characterization (Clavadetscher 1998), the open source community consists of a much more dynamic or ad-hoc user group. This is due to the fact that participants are free to contribute and free to leave the community. Therefore, sampling seems to be the most effective way to characterize open source user groups. Next, we classify users based on their years of development experience, motivation to contribute, and number of projects.

First, it is important to note that in general nearly 60% of the users have more than five years of software development experience, as shown in Figure 1. For users belonging to the core development team of an open source project other than the one they contribute to as users (referred to here as user group 1), the percentage is as high as 64%. Even with users who do not belong to any core development team (referred to here as user group 2), 41% of them have more than five years of development experience. Comparing this to the finding in (Zhao and Elbaum 2003), 61% of core developers have more than five years of development experience, it is interesting to note that the user profile is very close to the developer profile in terms of development experience. In other words, users who collaborate with developers are almost as experienced as the developers themselves. The user profile varies from project categories (Figure 2). The most obvious observation we can draw is that users of ‘database’ projects tend to be least experienced users and users of ‘software development’ projects seem to be most experienced in software development.

On average, 64% of users participate in open source development because of personal development needs, as shown in Figure 3. This percentage is very close to the 60% of projects that were started also due to personal needs (Zhao and Elbaum 2003). Roughly, this number

Table 1. Sampling distribution

<table>
<thead>
<tr>
<th>Project category</th>
<th>Number of users</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communications</td>
<td>151</td>
</tr>
<tr>
<td>Database</td>
<td>106</td>
</tr>
<tr>
<td>Games/entertainment</td>
<td>169</td>
</tr>
<tr>
<td>Internet</td>
<td>183</td>
</tr>
<tr>
<td>Multimedia</td>
<td>143</td>
</tr>
<tr>
<td>Software development</td>
<td>174</td>
</tr>
<tr>
<td>System</td>
<td>168</td>
</tr>
</tbody>
</table>
reflects the percentage of open source community members that were motivated by personal needs. This is consistent with (Hars and Ou 2001) and (Lakhani and Wolf 2003). Also interesting to note is that about 32% of the people participate because of the needs of their employers. This proves the fact that besides a few well-known open source products, more and more open source products are being adopted or sponsored by industry and organizations. When comparing user groups 1 and 2, it appears that group 2 users are motivated more by employer needs while group 1 users are more motivated in the other categories.

Another aspect we evaluated is whether users of different software categories tend to have different motivations to contribute. Figure 4 does indicate big gaps among different categories. For example, in the category of multimedia software, 24% of users contribute because of employer needs, but as many as 49% users in the Internet category are motivated by employer needs. A similar situation happens in the motivation ‘just for fun’.

Figure 1. Development experience

Figure 2. Development experience across categories

Figure 3. Motivation to contribute
Compared to users of communication software, users of software development are much less motivated by having fun.

As to the issue of how many projects a user is associated with, on average 40% of the respondents are involved in more than five projects, as shown in Figure 5. However, in general user group 1 is able to get involved in more projects than user group 2, especially under the category of more than five projects. This is possibly due to their higher level of experience and motivation, as illustrated by Figure 3.

COLLABORATIVE DEVELOPMENT

Unlike the practices of user collaboration in traditional software development, open source user collaboration carries unique characteristics such as taking on extensive development tasks, forging closer user — developer relationships and engaging in other active contributions. This section describes how users achieve such collaboration.

The majority of users inspect project websites fairly frequently — daily (29%) or weekly (52%). Compared to findings in Zhao and Elbaum (2003), 43% of the products are released every month and 29% every quarter, this indicates that users are able to receive updates in a timely manner even without release notifications from core development teams. This type of user activism or project information pulling can also be seen in Figure 6, which shows 79% of users get project updates by themselves. The next important source to get project updates from is community discussion (34%). Only 28% of users receive updates from developers.

The study also classifies the type of tasks users participate in and to what extent some of these tasks generate discussions among community members. Figure 7 shows the various types of tasks, sorted by percent of users. The top three ranked tasks are finding bugs, suggesting features, and finding usability problems. This is followed by tasks relevant to source code, including source code review and submission. Very few users offer administrative assistance to projects. Another interesting finding is that for group 2, the percentage of people is higher than group 1 users only in the category of finding usability problems. This finding suggests that a good mix of participants (not only experienced developers) is key to performing collaborative tasks evenly and to ensuring high quality.
As is evident from Figure 8, some numbers vary across different categories of software. The biggest variances occur in the category of ‘database’, in which a higher percentage of users find usability problems and the category of ‘software development’, in which a higher percentage of users tend to review and inspect source code.

Comparing it with Figure 2, the phenomena again highlights the importance of having a good mix of users with various development experience who can spontaneously and evenly distribute collaborative tasks. Overall, 64% of the respondents feel the collaboration is very effective and desirable.
USE OF COMMUNICATION MEDIA

When developers are geographically dispersed, their communication success may depend on utilizing effective and well-designed groupware (Nunamaker 1999; Deck and McHugh 2003) with achieving higher media richness as one of the design objectives. Generally speaking this is true. However, the open source development model somewhat defies this principal. High motivation plus extensive use of lean media such as email, discussion forum, version control tool, and trackers (for bug, patch, etc.) contribute to the success of such collaboration. We address the usage of these tools below.

As shown in Figure 9, most of the users (81%) use online forums to search for possible solutions to their problems. This is consistent with the findings of discussion archive usage, in which 92% of respondents indicated that they usually try to search for possible solutions in the archive before raising any question for discussion. This proves the fact that the online forum is becoming the primary knowledge base and collective group knowledge repository of the open source community. Another interesting finding is that group 2 users tend to show more interest in participating or soliciting forum discussions than group 1 users.

As to the use of version control tool (CVS), it can be seen that 55% of users check out source code using CVS for review or other purposes (Figure 10). Also, 27% use CVS to submit their own code. Compared to the fact that 53% of users did submit source code (Figure 10), it can be inferred that close to half of the users submit source code through channels other than CVS. The main reason is likely due to the discrepancy of submitting source code using CVS between user group 1 (32%) and group 2 (18%). This fact might also imply that users, especially those of group 1, may not be familiar with CVS system or CVS is not adequate enough for them to submit source code.

As shown in Figure 11, email is still a major communication tool for users to communicate with developers and receive messages from mailing lists (Yamauchi 2000). Apparently, more emails are used to communicate with developers than with other users. Also, receiving messages from mailing lists is another primary use of email. Respondents rank these communication tools by their importance as follows: mailing lists, discussion forum, trackers, version control tool, and personal emails (Figure 12).

COLLABORATION OUTCOMES

The collaborative relationship between open source developers and users is essentially a group problem-solving process, regardless of how people are dispersed geographically, or what motivations and accountabilities

![Figure 9. Online forum usage](image1)

![Figure 10. CVS usage](image2)
Figure 11. email usage

Figure 12. Ranking by importance

Figure 13. Issue prioritization

they undertake, or even how lean the communication media they use. Although it is too early to argue the appropriateness of lean communication media in accomplishing collaborative tasks, we can identify clearly some communication barriers that researchers try to remove by group support systems (Desanctis and Gallupe 1985) in open source user collaboration. For example, in Figures 13 and 14, we can see that voting rarely takes place in prioritizing issues and resolving conflicts. Most often developers take exclusive responsibility for such decision-makings. About 55% of the respondents agree that typically issues are prioritized based on developers’ preference. Only 12% of the respondents feel prioritizations were made by formal or informal voting. Thus, it is clear that developers are playing a much more influential role in handling issues raised by users. This type of group communication may not be consistent with the underlying social protocol that open source projects should take.

Even though respondents (57%) are very positive that developers take timely actions on user contributions and suggestions, the unbalance appears that developers are more interested in fixing bugs and flawed features than adding new features (37% against 16% as shown in Figure 15).

As to the benefits of contributing to open source development, apparently, the most popular use of open source software is to integrate them as components (64% of respondents) and acquire knowledge from participating in development (61% of respondents.) This result validates Ye and Kishida’s (2003) theory: ‘[w]e theorize that learning is one of the motivational forces.’ It is also interesting to see that source codes are really reused quite
often as reported by the 52% of the respondents who indicated that they reuse source code in their own software development (Figure 16). Again, the usage of open source also varies across different software categories. As shown in Figure 17, communication software gets the highest usage rate than other categories, while multimedia receives the least usage. Re-examining the motivations across software categories in Figure 4, it should not be a surprise since, in general, contributing to multimedia software is less motivated than other categories.

The study also aimed at finding the major collaboration difficulties among user group. Responses indicate that language is still a major communication problem since 24% of respondents choose the option of ‘hard to explain problems electronically’. In addition, at least six users pointed out difficulties of communicating with other people with different native languages. This problem also occurs when source codes are commented and/or documented by languages other than English.

**DISCUSSION AND CONCLUSION**

The earlier sections of the paper presented findings from the survey that gave quantitative evidence for understanding user collaboration in open source development. In this section, we summarize and discuss implications that can aid future developments of collaborative environment and define relevant management strategies for the open source community.
Experienced users

Open source users are already well recognized for their active participation in development. However, their actual level of development experience and capability to contribute to multiple projects is beyond our expectation. The positive side of this fact is that the ‘virtual open source development teams’ are actually having contributors who are comparable to core developers in terms of development experience. On the other hand, since the findings also point out that users with less development experience are willing to take more responsibilities in certain collaborative tasks, such as finding usability problems, it is very important for the open source community to pay attention to this type of user. This should include offering a better environment not only to open ‘sources’, but also ‘executables’ to them since without their inputs, a technically perfect software may not be acceptable by other users.

Communication support

As a distributed group problem-solving process, open source still relies on lean media to achieve collaborative tasks. It is true that factors like incentive, experience and extensive lean media usage favor the success of such collaborative development. Nonetheless, these factors do not completely remove some barriers of group problem solving. For instance, a few core developers still play dominant roles in resolving conflict issues or prioritizing issues, which might not be necessarily desirable for the users or even developers themselves. Although we can expect to solve such problems using group support systems, the right approach is not to carelessly introduce any existing groupware or group support system into open source development. If the tasks do not fit the technology, the groupware system will not work effectively (Zigurs et al. 1999). As the first step of achieving such fitting, our survey has already provided useful information on tasks conducted in the user collaboration.

Knowledge base

Open source is becoming a knowledge base for people to acquire relevant information and skills. People participate in community discussions not only to improve the software, but also to learn from participation. Unlike traditional settings, this type of learning occurs across various projects as one user may participate in multiple software development projects. The open source development model truly opens up a completely new area of research for knowledge management and for the success of knowledge management in terms of affecting users’ enthusiasm and effectiveness of further contribution. Generally, the present public open source hosting sites are still lacking good mechanisms specifically designed for this kind of learning purposes, especially cross-project learning.

Language barrier

Due to the special nature of open source, collaborating users come from nearly every place around the world. The fact that computers could offer better communication abilities, and what mechanisms to use, to solve language and culture barriers is important as well. Computer-based communication system should be able to, for example, facilitate a ‘user-to-user’ assistance by delivering language translation help for users at an acceptable responding speed.

Commercial vendor support

Recently, a growing number of companies, such as Sun Microsystems, Apple and IBM, began to release their software as open source to take advantage of the enormous benefits of voluntarily user collaborative software development. (Glass 2003) points out that, considering the ‘with enough eyeballs, all bugs are shallow’ quote, if
not enough eyeballs are devoted to a software product, there is every reason to expect that such open-source software may be less reliable than it should be. Therefore, the advantage of open source can be materialized only if a sufficient number of users join the development. Our survey findings inform commercial vendors that prior to engaging in open source projects, it is useful to evaluate relevant factors, such as learning opportunities and category of software to be developed, because it is reasonable to expect significantly different amount and type of user contributions under different circumstances.

Overall, open source is no longer focused on the simple idea of releasing source code, nor is it a concept of getting others involved in modifying one’s source code. The open source development model has become a complex paradigm for gathering a large number of experienced and sophisticated users to participate in software development together with core development teams. Our study provides succinct results for understanding the user collaboration side of the open source development model. There is still much more to learn about this paradigm and how to better support it, which is already in our on-going and future research directions.

Finally, it is important to mention several factors may cause biases in the results of such studies. Although SourceForge is the largest open source project host site, there are many others that should be included in a more comprehensive survey. The response rate is always an issue in research studies. The relatively low response rate here might be the consequence of two reasons: the complexity of the questionnaire and computer accessibility of casual open source users.

ACKNOWLEDGEMENT

The authors wish to thank all the open source users who took the time to respond to our survey.

References


APPENDIX: QUESTIONNAIRE

Part I: About yourself

1. Your software development experience
   A. None
   B. < 1 year
   C. 1–5 years
   D. > 5 years

2. Are you a developer of any open source projects?
   A. Yes
   B. No
   (If yes, please answer remaining questions based on your contributions to other open source software of which you are not a developer.)

3. What is your motivation to participate and contribute to open source?
   A. Just for fun
   B. To meet personal development needs
   C. To meet employer needs
   D. To meet needs of my open source project(s)
   E. Other

4. How many projects are you involved in?
   A. 1
   B. 2
   C. 3
   D. 4
   E. 5 and up.

Part II. About collaboration

5. How often do you visit project web pages?
   A. Daily
   B. Weekly
   C. Monthly
   D. Other (  )

6. What do you do for the open source projects?
   A. Find bugs
   B. Find usability problems (e.g. ugly user interface, compiling, installation inconvenience)
   C. Suggest new features
   D. Review and inspect source code
   E. Submit my own source code
   F. Offer project administration assistance
   G. Documentation
   H. Other (  )

7. Please RANK (e.g. BCDEA) the following non-technical-support topics based on their popularity in online group communications
   A. Bugs discovered at run-time
   B. Bugs discovered in source code
   C. Usability problems (e.g. ugly user interface, compiling, installation inconvenience)
   D. Adding/removing features
   E. Other (  )

8. How do you keep yourself updated with project progress?
   A. I receive updates from development team periodically
   B. I check out project progress (e.g. new releases) by myself periodically
   C. I receive such information primarily from community discussions
   D. I am not kept updated
   E. Other (  )
9. How do you feel the effectiveness of solving open source problems as a group of both developers and users?
A. Very effective and desirable
B. Effective but not so obvious
C. Ineffective
D. Not applicable

Part III. About electronic communication media

10. Do you use online forum to
A. Solicit discussions to issues I raised
B. Discuss issues raised by other people
C. Search for possible solutions to my problem
D. Track development progress
E. Other ( )
F. I rarely use it

11. Do you search communication archives before raising or discussing any issues?
A. I usually do and often find useful information there
B. I usually do but rarely find useful information there
C. I rarely do
D. Not applicable

12. Do you use version control tools (e.g. CVS) to
A. Track development progress
B. Check out source code for review or other use
C. Submit my code
D. Other ( )
E. I rarely use it

13. Do you use structured trackers (e.g. Bug tracking, RFEs and Patch submission tools) to
A. Track development progress
B. Track stuff I submitted
C. Give feedbacks on stuff other people submitted
D. Other ( )
E. I rarely use them

14. I use e-mail to
A. Communicate with developers
B. Communicate with other users
C. Receive messages from the Mailing list
D. Other ( )
E. I rarely use it

15. Please RANK (e.g. BCDEA) the following communication tools based on their importance
A. Discussion forum
B. Mailing lists
C. Version control tool (CVS)
D. Structured trackers (e.g. Bug tracking, RFE, Patch submission tools)
E. E-mail

Part IV. About collaboration outcomes

16. How issues raised by different people are prioritized for solutions?
A. We vote for them formally or informally
B. It depends on developers' preferences
C. It depends on how much discussion is received on a specific issue
D. Other ( )

17. How conflicts on opinions (or submissions) are usually resolved?
A. We vote for them formally or informally
B. Usually developers come to resolve conflicts
C. Usually experienced users come to resolve conflicts
D. Conflicts are rarely resolved
E. Other ( )

18. How do you describe developers’ follow-up on user contributions
   A. Developers often take actions on issues raised by users in a timely manner
   B. Developers are more interested in adding suggested new features
   C. Developers are more interested in fixing bugs and other misfeatures
   D. Developers are not responsive to user contributions
   E. Other ( )

19. Please identify any of the following communication problems pertaining to you
   A. It is difficult to explain complicated software problems electronically
   B. I have problems communicating with developers because of background difference
   C. I have problems communicating with other users because of background difference
   D. The informal discussion style is very confusing and messy
   E. Other ( )
   F. Not applicable

20. If you are a software developer, how participating in open source community benefits your software development
   A. I reuse source code from open source projects
   B. I integrate open source software to my system as component(s)
   C. I use open source software as a template to start building my system
   D. I acquire relevant knowledge from community communications
   E. Other ( )
   F. Not applicable