What Type of Thread Can Get Feedback in OSS User Mailing List?

Akinori Ihara, Yuji Tsuda, Ken-ichi Matsumoto
Graduate School of Information Science, Nara Institute of Science and Technology
8916-5, Takayama, Ikoma, Nara, JAPAN 630-0192
{ akinori-i, yuji-tsu, matumoto } @ is.naist.jp

ABSTRACT

High quality Open Source Software (OSS) has been used in many commercial software developments. In order to provide technical support to end users, OSS projects manage a user mailing list. It is for discussion about bugs and new functions of OSS with end users. However, according to a survey of Japanese companies that use OSS to build their commercial software, the biggest problem is the lack of adequate technical support. In this study, we investigate what type of thread can get feedback in user mailing list. As a result of a case study using Apache and Python project data, a thread is posted by a deep experienced user would be received a useful answer. In addition, we found threads written about internal system information of the OSS is more likely to be replied.

Categories and Subject Descriptors

D.2 [**Software Engineering**]: Management; D.4.4 [**Communications Management**]: [Message sending]

General Terms

Management

Keywords

Open Source Software, User Mailing List, Communication

1. INTRODUCTION

Recently, many software companies use Open Source Software (OSS) such as Apache, Linux to build their development environment and to create a commercial software. The software companies hope to be short-cycle development, to reduce development cost. On the other hand, some of the company developers concerned about the quality and the maintenance. According to a survey of IPA [4] in 2009, the most key disadvantage of OSS for software company developers is difficult to get technical supports by OSS project in case of an emergency. If companies have troubles about OSS, they ask questions in user mailing list provided by OSS project to get supports.

In this study, we investigate when and what should the companies' developers post a thread to user mailing list to get technical supports from OSS project. The user mailing list is used by not only users but developers use it to get some feedback (new function requirements and bug reports) from users directly [7]. If we post a thread to the user mailing list, developers or users may reply. On the other hand, unfortunately you may not get replies. Using Apache and Python project data, we reveal what type of thread can get feedback in user mailing list in terms of the feature of sender, posting time, and message context.

2. MAILING LIST IN OSS PROJECTS

When users get information about OSS, they search in web sites at first. Next, they search in a mailing list [3], because large-scale OSS projects manage various mailing list to share information such as development, bug fixing, and trouble (e.g., developer mailing list, user mailing list).

In user mailing list, the participants are users and developers. Some users have a deep knowledge of the OSS. They exchange information each other in the user mailing list. However, according to survey of Japanese companies use OSS to build their commercial software, some of them have not yet provided the technical support in user mailing list. We consider that the user mailing list is not working well. In this study, we analyze what type of threads can get reply in user mailing list in terms of sender and message context that Dabbish et al. presented important factor influenced users' perceptions [2]. Also, we focus on posting time.

Sender: If a user has used the OSS for a long time, he may get a feedback from the other participants in user mailing list, because he may send a clear message to user mailing list. Also, if a user often post a thread to user mailing list, he may get a feedback from participants who communicated with him in the past. On the other hand, if a user has never sent e-mail to user mailing list, he may be difficult to get feedback from participants in the mailing list.

Posting time: Geographically-distributed developers and users use OSS. Some of them at day or night, on weekday or weekend [8]. Therefore, they are difficult to contact to the other people in real time each other [6]. And they do not know when they should post a thread.

SSE'13, August 18, 2013, Saint Petersburg, Russia
Copyright 2013 ACM 978-1-4503-2313-0/13/08...$15.00
http://dx.doi.org/10.1145/2501535.2501541

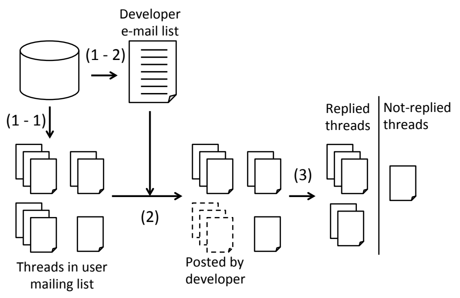

Figure 1: Method to extract the treads from mailing list archive.

Message context: Many messages are posted to a user mailing list on a daily basis. Developers do not only read the message, but also they have to fix bugs, expand new functions, and write many documents. They cannot spend enough time to read the messages in user mailing list. Then, they may decide which message they should read based on message context (e.g. the characteristic words, writing skill) to save time.

In this study, we analyze what type of messages will be replied based on these point.

3. ANALYSIS METHODS

3.1 Message Parsing

In our study, we analyze e-mails collected from user mailing list archive provided by OSS project. As the collected messages were ordered in time, we need to classify the e-mails into the same discussion (thread) to know whether the e-mail did get the replies or not. If nobody replied, we consider that sender could not get feedback from the OSS project. Figure 1 shows our approach from collecting e-mails to classifying threads into the replied threads and the not-replied threads.

(1) Collecting e-mails from mailing list archive
OSS projects provide mailing list archive in their web page. From the archive, user mailing list and developers mailing list are collected. In the user mailing list, not only users but also developers post messages for announcement. Since our goal is to know whether users can get feedback in user mailing list, we remove threads posted by developers.

(1-1) Classifying e-mails into same discussion in user mailing list
Using a user mailing list, the e-mails are classified into threads. When users or developers reply to a posted threads, most of them do not change the subject of the e-mail, because the other participants easily find the discussion in the future. In our study, we classify the e-mails into threads based on the subject.

In addition, some users or developers use more than one e-mail address by oneself. If different e-mails were posted by different e-mail and same name, we unify the e-mail address. Also, a user or a developer use different name between a couple of e-mail systems. In such case, we unify the name.

Table 1: Summary of the target data

	Apache	Python
target period	2001/11-2013/2	1999/5-2008/11
# of messages	90,004	522,282
# of threads	32,572	99,777
by developers	6,467	17,364
by users	26,105	82,413
replied	13,981	60,090
not replied	12,124	22,323
# of users	10,727	28,399

(1-2) Extracting developers' e-mail from developer mailing list
To remove the threads posted by a developer from user mailing list, developers e-mail addresses are collected from a developer mailing list. And developers' e-mail address list is created.

(2) Removing threads posted by a developer
Threads posted by a developer are removed from the threads collected in (1-1) based on the developer e-mail address list.

(3) Classifying threads into replied threads and not-replied threads
Threads posted by users are classified into replied threads and not-replied threads. The replied threads means that a thread posted by a user was replied by the other participants. On the other hand, the not-replied threads means a thread was not replied.

3.2 Analysis of Sender

Many participants often leave OSS project. A few active participants join an OSS project for a long time. On the other hand, many participants leave the project in a short period. The more participant have experience of posting to user mailing list, the higher the participant may be likely to get feedback. In this study, we calculate the rate of the replied messages according to sender who posted many threads and the number of posted threads.

3.3 Analysis of Posting Time

Most of developers posted e-mail in the evening and by night after their work [6]. However, there has been no study that tried to analyze when users post a thread to user mailing list and how many threads get feedback. In this study, we calculate the rate of replied messages according to week and hour.

3.4 Analysis of Message Context

Anybody can post a comment to user mailing list. The quality of comments depends on authors, because all users do not have a deep knowledge like developers. Therefore, a readable message may be replied. In this study, we calculate the rate of replied message according to the message context using tf-idf technique after stemming and removing stop words.

4. CASE STUDY

This section describes a case study which has been conducted to understand what type of threads can get feedback in a user mailing list. In our case study, we target two OSS projects: Apache HTTP Server project and Python Project.

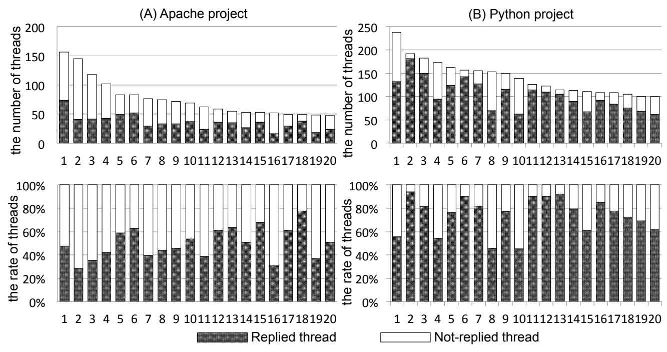

Figure 2: The number/rate of the threads acceding to senders. (left: Apache, right: Python)

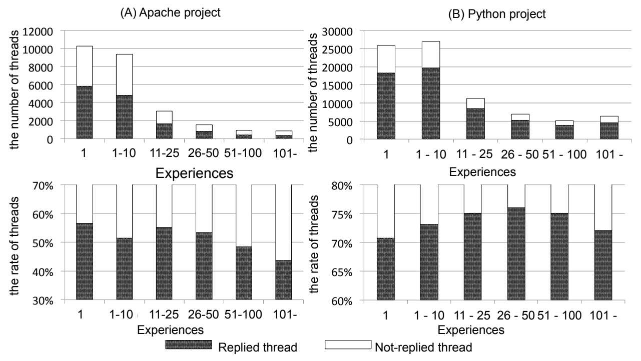

Figure 3: The number/rate of the threads acceding to experiences. (left: Apache, right: Python)

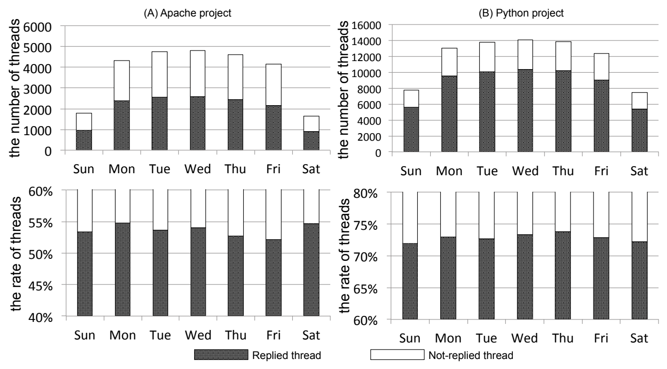

Figure 4: The number/rate of the threads acceding to week. (left: Apache, right: Python)

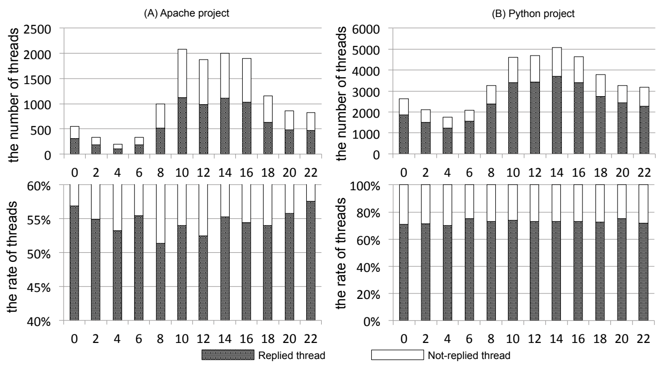

Figure 5: The number/rate of the threads acceding to hour. (left: Apache, right: Python)

4.1 Target Projects and Data

In the Apache HTTP Server project and the Python project, mailing list is used to discuss and share information about OSS development by users basicly. Apache HTTP Server Project has managed a user mailing list since Nov.2011. Python Project has managed a user making list since May.1999.

Table 1 presents the target period, and the number of message in each project. The number of threads posted by users is 26,105 (80% of all threads) in Apache project and 82,413 (83% of all threads) in Python project. In the threads posted by users, the number of the replied threads is 13,981 (54% of threads posted b y users) in Apache project and 60,090 (73% of threads posted by users) in Python project. Many threads in both of projects were not replied. This result clearly shows that the most key disadvantage of company developers is difficult to get technical supports by OSS project in case of an emergency. In our paper, we analyze the reason for not being replying in terms of sender, posting time and message context qualitatively and quantitatively.

4.2 Analysis of Sender

4.2.1 Replied threads according to sender

Figure 2 presents the number/rate of the replied threads (y-axis) posted by top 20 users (x-axis) in Apache project and Python project. The rate of the replied threads is 28%-77% for Apache project, 45%-94% for Python project. Looking at these results, we can see a difference between users. Hence, the rate of the replied threads may depend on message contents or an individual writing style.

4.2.2 Replied Threads According to the Number of Posted E-mails

Figure 3 presents the number/rate of threads (y-axis) according to the experiences (the number of the posted threads) of users (x-axis). In both projects, threads posted by users that have posted more than 50 times may not be replied. The rate of the replied threads drops by 10% over 100 times in Apache project. Also, it drops by 4% over 100 times in Python project. Looking at these results, we found that we should post a thread 11-50 times.The more participant posts threads, the less the threads are replied.

4.3 Analysis of Posting Time

4.3.1 Replied Messages According to Week

Figure 4 presents the number/rate of threads (y-axis) according to week (x-axis) in Apache project and Python project. In both projects, the number of threads that posted in the weekend is less than in the weekday. However, no matter what day of the week, the rate of replied threads was not so different between the weekday and the weekend. From this result, we found the thread was supported by geographically distributed developers or users.

Table 2: Differences of the high tf-idf score rank in the replied/not-replied thread in Apache project.

Higher replied thread rank

word	Replied thread	Not-replied thread
virtual	9	23
host	10	18
log	13	21
directory	14	25
redirect	15	34
access	16	24
rewrite	18	33
mod_rewrite	20	27

Higher not-replied thread rank

word	Replied thread	Not-replied thread
configure	10	21
request	12	32
install	15	22
module	19	33

Table 3: Differences of the high tf-idf score rank in the replied/not-replied thread in Python project.

Higher replied thread rank

word	Replied thread	Not-replied thread
list	5	24
newbie	8	19
string	9	51
class	10	21
function	11	25

Higher not-replied thread rank

word	Replied thread	Not-replied thread
install	12	20
embed	19	43

4.3.2 Replied Threads According to Hour

Figure 4 presents the number/rate of replied threads (y-axis) according to local time (hour) (x-axis) for the location where the sender stays in Apache project and Python project. In Python project, the rate of the replied threads drops early morning and at meals. On the other hand, in Apache project, the rate of threads was not so different between in a day. The rate of the replied thread in Apache project is lower than Python project in a day.

4.4 Analysis of Message Context

Table 2 and table 3 present high tf-idf score rank words in the subject of the replied threads or the subject of the not-replied threads, and the tf-idf score rank of a word in the replied threads more than 7 ranking different from the not-replied threads. We targeted only subject that is summarized the body of an e-mail.

Many internal system words (module name, log, internal function name) were written in the replied threads. On the other hand, many words for implementation (e.g. install, input command) were written in the not-replied threads.

5. DISCUSSION

From the result of our case study, we found 2 types of threads that are likely to be gotten feedback in user mailing list.

The number of the posted threads would be interesting to receive a useful answer.
According to our analysis of sender, threads posted by a user who have posted a couple dozens times are likely to be replied. We consider if users submit many messages, they may find partners who answer their questions, because OSS developers often contact to same developers [5].

Threads written about internal system information of the OSS is more likely to be replied.
When Bettenburg et al. analyzed what makes a good bug report, they got similar results [1]. They found when a user submits a bug report and reports code examples and test cases, developers are likely to fix the bug. As in the case of the bug report, if users post a message using the specific word such as module name, they may get a feedback.

6. CONCLUSION AND FUTURE WORK

In this study, we analyzed when and what should users send e-mail to user mailing list in OSS project. Using Apache and Python data, we conducted a case study to understand how many e-mails can get feedback in the user mailing list in terms of sender, posting time, message context. We found that users should have experiences in a user mailing list to get technical supports. And, users post a message using the specific word such as module name.

Many threads are posted to user mailing list on a daily basis. Developers and users are difficult to check all of them. Users should understand this condition. If users use OSS and are also interested in OSS development, users should study how to write a clear e-mail. In the future, our study will indicate how to write e-mails to get feedback in user mailing list based on this study.

7. ACKNOWLEDGMENTS

We appreciate the anonymous reviewers giving insightful comments and helpful suggestions. This research is conducted as part of Grant-in-Aid for Young Scientists (B), 25730045 and for ScientificResearch (B) 23300009 by Japan Society for the Promotion of Science (JSPS).

8. REFERENCES

[1] N. Bettenburg, S. Just, A. Schröter, C. Weiss, R. Premraj, and T. Zimmermann. What makes a good bug report? In *Proceedings of the FSE'08*, pages 308–318, 2008.

[2] L. A. Dabbish, R. E. Kraut, S. Fussel, and S. K. Kiesler. Understanding email use: Predicting action on a message. In *In Proceedings of the CHI'05*, pages 691–700, 2005.

[3] K. Fogel. *Producing open source software: how to run sucessful free software project.* O'Reilly Media, Sebastopol, CA, 2005.

[4] Information Technology Promotion Agency. A survey report on open source software based businesses. (year 2009 edition), 2010.

[5] G. Jeong, S. Kim, and T. Zimmermann. Improving bug triage with bug tossing graphs. In *Proceedings of the ESEC/FSE'09*, pages 111–120, 2009.

[6] M. Ohira, K. Koyama, A. Ihara, S. Matsumoto, Y. Kamei, and K. Matsumoto. A time-lag analysis for improving communication among oss developers. In *Proceedings of the KCSD'09*, pages 49–62, 2009.

[7] E. S. Raymond. *The cathedral and the bazzar: musings on linux and open source by an accidental revolutionary.* O'Reilly and Associates, Sebastopol, CA, 1999.

[8] G. Robles and J. M. Gonzalez-Barahona. Geographic location of developers at sourceforge. In *Proceedings of the MSR'06*, pages 144–150, 2006.