A Method for Measuring OSS Projects' Activity based on Cluster Analysis of Email Threads

Kimiharu Ohkura[1] Yoji Onishi[1] Shinji Kawaguchi[1]
Masao Ohira[1] Hajimu Iida[1] Ken-ichi Matsumoto[1]
1 Graduate School of Information Science,
Nara Institute of Science Technology
8916-5 Takayama-cho, Ikoma-shi, Nara, 630-0192 Japan
E-mail: {kimiha-o, yoji-o, kawaguti, masao, matumoto}@is.naist.jp,
iida@itc.naist.jp

Abstract. In open-source software (OSS) development, activity is one of important factors which affect quality of software products. Various development tools' logs have been used to analyze certain OSS projects because of the lack of development documents in OSS projects. However, it is difficult to analyze OSS projects in common since tools used in software development vary from project to project. In this paper, we focus on mailing lists (MLs) that are used in most OSS projects, and propose a method for activity prediction based on in co-occurrence of developers in MLs. From the result of our experiment, we confirmed that the co-occurrence of developers affected the activity in OSS projects.
Keywords. Cluster Analysis, Mailing list (ML), Activity, Project Analysis

1. Introduction

As a form of software development, open source software (OSS) development has much attracted researchers' and practitioners' interests recently. OSS development is mainly performed via some networks such as WWW. Mailing list (ML) is widely used as a communication tool among developers or users in OSS projects. Unlike corporate software projects, OSS projects do not necessarily need to record operation logs or documentations and to keep source codes confidential. Furthermore, OSS developers are voluntary to participate in/drop out of a project. Therefore, the burden of responsibilities imposed on OSS developers is less than that of corporate developers.

While OSS projects allow developers to take less responsibility, however, they often face the following problems. (1) It is apt to lack information which is necessary for a postmortem or for understanding the progress of a project, and (2) it is hard to analyze OSS projects due to standardized data to be analyzed. Activities in OSS projects are an important factor which has an effect on the quality of OSS products. So it is desirable for project managers or organizations which use OSS products to precisely know the activities, but we can grasp the activities only in a limited way from the above reasons.

In this paper, we focus on ML that is used in most OSS projects, in order to grasp activities in OSS projects and then we propose a method for analyzing activities in

OSS project based only on ML archives. We define *activity* as the degree of vitalization of a software project. “High activity” means that the project is in good progress. By contrast, “Low activity” means that the project is stagnant. We framed a hypothesis based on the activity as follows.

Hypothesis: Higher activity in developers’ ML means lower co-occurrence of the developers

We think that discussions would fail to reach agreement, if long threads posted by many developers are large in number as a consequence of the sluggish discussions, thus we consider that the development is stalled. In this paper, we test the hypothesis based on calculating the co-occurrence of the developers by cluster analysis. Section 2 shows related work, and we describe a proposal analysis method in section 3. In section 4, we test the hypothesis by an experiment. Finally, conclusion and future work is in section 5.

2. Related Work

Recently, many researchers have been focusing on exploiting the repositories used in OSS development (e.g., version management system and bug tracking system), in order to complement the lack of information mentioned Section 1 and then to better understand the software development in OSS projects [1] [2] [3]. Then, recent studies on OSS projects have suggested that OSS projects needed further developers and participants’ active contributions to the OSS development. For instance, Mockus et al. [4] reported only 4% of developers in the Apache project created 88% of new code and fixed 66% of defects. From a total of 196 developers in the Ximian project, 5 developers account for 47% of the modification requests (MRs), while 20 account for 81% of the MRs, and 55 have done 95% of them [5]. Beenen [6] reported that projects with a large proportion of non-contributors have difficulty providing needed services such as bug fixes and software enhancements to all the members. Hanakawa et al. [7] also proposed a tool for supporting project management based on bug report data. Although these studies have partially revealed the activities in OSS projects by analyzing the data from the repositories described above, it is so difficult to apply the analysis method or tools to other OSS projects since each project often uses different repositories, that is, there is no standardized data to be analyzed. In this paper, we propose an analysis method using the archive data of ML which is widely used in most of OSS projects. We believe that our method would be applicable to analyze activities in OSS projects.

There also exist studies targeting on the ML data to analyze OSS projects. For instance, Yamauchi et al. [8] described how geographically distributed developers coordinated their activities and how electronic media was used in the coordination, by analyzing mailing-lists of FreeBSD Newconfig and GNU GCC project. Bird et al. [9] examined the centralities of Apache developers from the developer mailing-lists, based on social network analysis (SNA) Rigby et al. [10] reported the linguistic characteristics of excellent developers and the changes of words and actions of

developers before/after a major version of software released. Though using the ML archive data to analyze activities in OSS projects is similar to our approach, the analysis methods used in the studies above requires analysts to have specialized knowledge or to analyze the ML data by a particular format (e.g. a body of a message must be written in formal English, and so on.). The proposed method in this paper is applicable to OSS projects which use ML to communicate among participants and analysts do not need to have a special kind of knowledge for analysis.

3. Method

This section describes our analysis method for measuring the co-occurrence of OSS developers from ML archives. Our method consists of the following 4 steps.

1. Extraction of unique users: Unique users are extracted from the ML archives.
2. Extraction of email threads: Email threads are reconstructed from the emails' header.
3. Thread encoding: The threads are vectorized based on frequencies of messages posted by each user.
4. Clustering: Vectors converted by the encoding are classified into clusters using the hierarchical clustering algorithm.

We describe detailed procedures in the following.

3.1. Extraction of Unique users

First, unique users participated in the ML are extracted. The users are identified by a line started with "From:" in the email header. We preprocessed the ML archives to identify unique users as follows.

3.1.1 Dividing a From-line into Name and Address

A from-line consists of a name part and an address part surrounded by quotes. The following is an example of the division.

From: Kimiharu Ohkura <kimiharu@is.naist.jp>
→ Name Part : Kimiharu Ohkura
→ Address Part : <kimiharu@is.naist.jp>

If the same address is used by users with different name parts, we consider they are the same person.

3.1.2 Removal of White-spaces/Symbol Characters

We also attempted to remove symbol characters and white-spaces from name and address part. White-spaces include space, tab and linefeed. This is because

unnecessary added quote or white-spaces are existed in a name and address part. For example, following two names are recognized as different users without symbol removal.

Name1: Ohkura
Name2: "Ohkura"

3.2. Extraction of Email threads

In order to extract email threads, all messages are reconstructed as tree structure on memory referring to an "In-Reply-To:" and a "Message-Id:" header. Although "Reference:" headers are normally used to extract the threads, it was no need to regard in current case because there were very few emails contained the reference headers, and the email tree could be constructed using only the reply headers.

The preprocessing mentioned above was processed email archives in MBOX format by Perl script we made for this study. There are some corrupted data such as messages with duplicated message id, or body text included line head starts with "From". They were processed manually.

3.3. Thread Encoding

Next, we encode the threads based on a list of unique users and participants in each thread. This encoding is executed through vectorization by recording post frequencies of each user (See **Fig. 1**). After all threads were vectorized, distances among the threads are calculated based on the Vector Space Model [11].

	Developer A	Developer B	Developer C	Developer D	Developer E
Thread 1	0	2	1	1	0
Thread 2	0	0	0	1	1
Thread 3	2	1	1	0	0
Thread 4	0	1	1	0	0
Thread 5	0	1	0	1	1

Fig.1 An example of vectorization by frequencies of messages posted by each user

Cosine similarity that is a measure of distance using intervector angle was used in this research. Other measure that commonly used is a Euclidean distance. However, the measure is unreasonable in this study because it has possibility that regards no co-occurrence vectors (same as two vectors that product set is empty) as similar. Cosine similarity is based on inner product, and distances of no co-occurrence vectors are calculated to be 0. It means that similarities among no co-occurrence vectors cannot be calculated as high degree (i.e. small distance), and sound results are obtained.

In this study, we assume that similar threads are contained similar participants because of using the post frequencies as feature.

3.4. Clustering

Finally, we classify the threads based on clustering algorithm. The group average method, one of the hierarchical algorithms, is used in this study. In this method, a distance between clusters are defined as an average of distances among each clusters' elements. Then, it merges clusters in order of increasing the distance.

Generally, the Ward method is widely used as a high accuracy algorithm. However, the method cannot be used because the Ward method is premised on Euclidean distance but we use the cosine similarity as a distance measure. Therefore, the group average method, considered that high accuracies are obtained is used by our clustering in this study.

4. Experiment

We conducted an experiment for validation the hypothesis mentioned in section 1.

4.1. Dataset

The target of our experiment is the data from Kaffe [12] which is a development project of OSS Java VM started in 1995. CVS repositories and ML archives in the project has been open to public since 1996. We applied our analysis method to the ML archives, and confirmed validity by investigation of correlations between the project activity and co-occurrence of the developers.

4.2. Methodology

We applied our analysis method to the ML archives of the target project month by month. Then some statistical values about the result of clustering ware compared to CVS commit graphs that represented the project activity in each month. This experiment was conducted on the project data from January 2004 to December 2005.

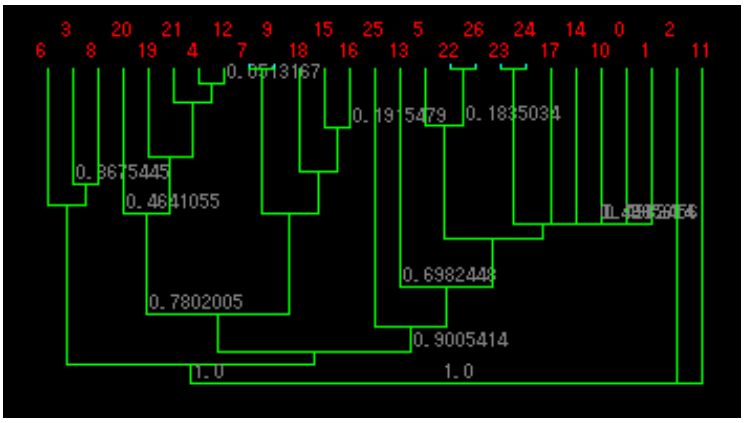

Fig.2 An example of a dendrogram based on the hierarchical clustering algorithms

In order to obtain valuable results by the hierarchical clustering method, it needs a proper threshold as a dividing point because the method finally merges all the child nodes into one cluster (i.e. a root of tree structure) without a particular threshold. **Fig. 2** shows an example of the result of the clustering represented as a dendrogram. Although the number in Fig. 2 approaches 1.0 toward the upper hierarchy due to convenience of our program drawing dendrogram, the cosine similarity is normally defined that approaches 1.0 toward the lower hierarchy. In other words, a high similarity indicates a short distance. By the same token, we refer to "high similarity" as "short distance among/between the vectors" in this paper.

We derived the threshold contained the sufficient co-occurrence is 0.5 based on our preliminary experiments.

4.3. Result and Discussions

We compared commit graph with some statistics about the clusters including average, variance, maximum value, and the number of cluster measured by the method mentioned in section 3 month by month.

First, we discuss the statistics about the clusters' elements in the result. **Fig.3**, **4** and **5** show commit graphs, compared to the statistics (average, variance, and maximum value). The bar chart indicates the number of commits. On the other hand, the line chart indicates each statistic.

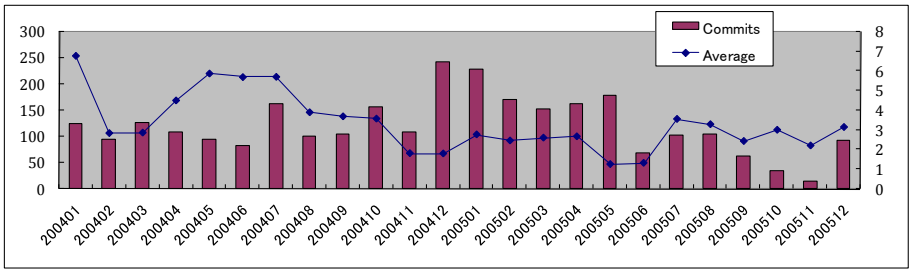

Fig.3 The transition of the average and the number of the commits

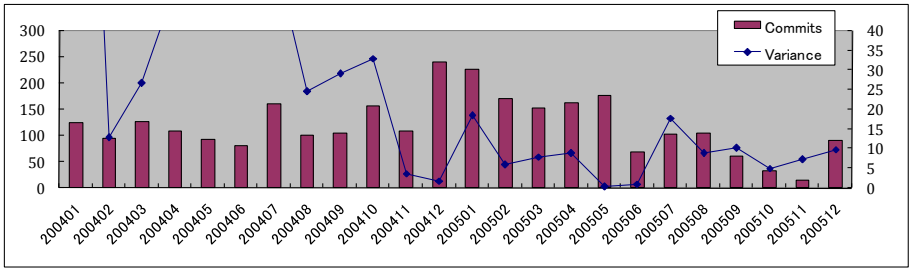

Fig.4 The transition of the variance and the number of the commits

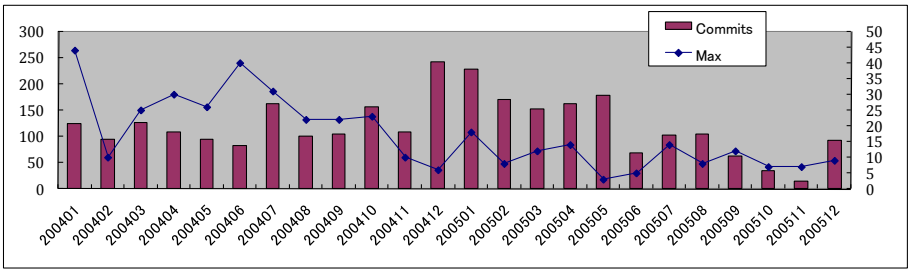

Fig.5 The transition of the maximum value and the number of the commits

The average (Fig. 3) and the maximum value (Fig. 5) have a tendency to fluctuate widely in 2004 and the fluctuation gradually weaken toward the latter part of 2005. However, the variance (Fig. 4) does not have a significant tendency.

The average indicates mean topic scale of each month. Larger mean topic scale represents that many participants join many topics, which has high co-occurrence. Therefore, higher average of the clusters' elements indicates sluggish development if our hypothesis is true. The figure reflects the fact that high degrees of the commits were recorded during the end of 2004 to June 2005, a period of low average. It is seemingly consistent with the hypothesis. However, the clusters' elements are strongly affected by the number of emails sent to the ML during the month because the elements are email threads.

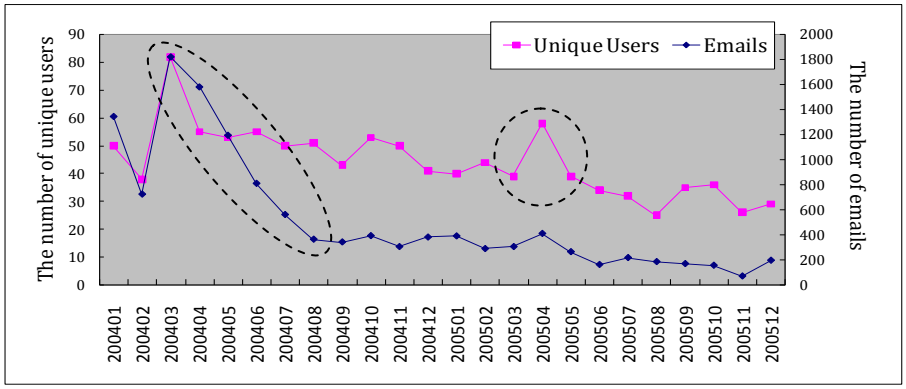

Fig.6 The transition of the number of the unique users and the email sendings

Then, we inspected a transition of the number of emails sent during the period in this experiment. The result is shown in a part of the **Fig. 6**. The figure indicates that email sendings considerably declines from March 2004 to August 2004. Because the average of the clusters' elements indicates a similar tendency from August 2004, we cannot say the average value correlates directly with the project activity.

The maximum value of the clusters' elements indicates a scale of the largest topic of each month. A higher degree of the value is obtained when larger scale topics such as a questionnaire for all developers or discussion about some policies after a major release. By contrast, a lower degree is recorded when only small topics occurred during the month. However, in the result of this experiment, the maximum value is not much different of the average. Therefore, it is difficult to analyze the project activity from this result.

As mentioned above, we can conclude that the statistics of the clusters' elements cannot be used to measure the project activity because they are affected by the number of emails/email threads.

In the later part of this section, we discuss not the number of the clusters' elements but the number of the clusters by comparison with commit graph of each month. A cluster is represented as a group containing some topics in which similar participants tend to join. In this way, a cluster that has only one element is counted, thus fewer clusters indicate that the project has a lot of co-occurrences during the month. The transition of the number of the cluster is shown in **Fig. 7**. According to our hypothesis, if there are many clusters, the co-occurrence is low. That means the project has higher activity during the month.

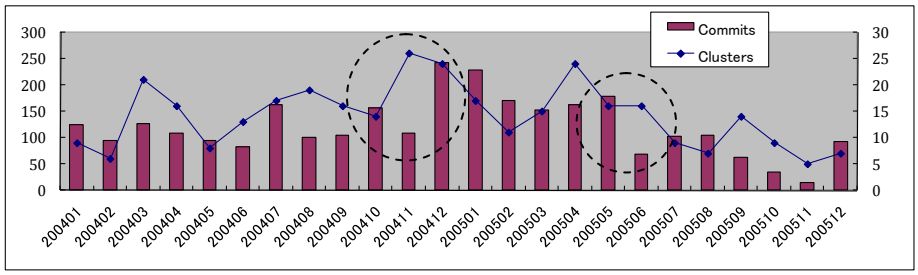

Fig.7 The transition of the number of the clusters and the commits

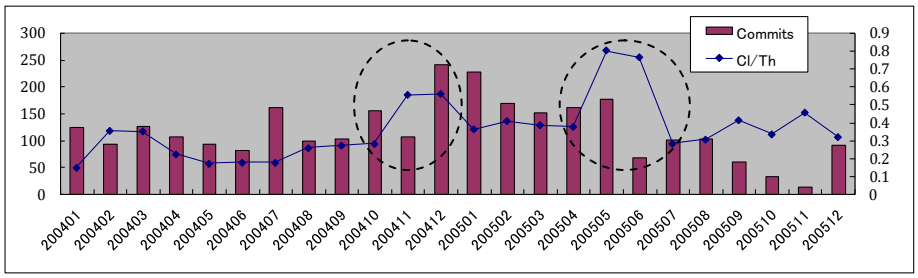

Fig.8 The transition of the cluster rate and the number of the commits

We focused on the sudden increase of the commits from November to December 2004 and the sudden decrease of the commits from May to June 2005. We think the increase indicates that the development is taking off because of the high activities as shown by the co-occurrence. Likewise, we attribute the increase of commits next month or later to increase the activities. However, a short time later, the graph shows a downward tendency toward February 2005. Although it increases again at April 2005, we think this phenomenon does not have correlation with the activities due to long gaps between the phenomenon (April 2005) and June 2005, the period of the sudden increase.

In response to the result, we focused on the number of unique users (unique senders) in the MLs of each month. The dimension number of the vector is increased by the number of unique users, thus the clusters tend to be subdivided into smaller topics. The transition of the number of unique users in the target project from 2004 to 2005 is shown in a part of Fig.6 as a line chart. The number of unique users at April 2005 is large. We think the phenomenon is caused by the fluctuation at April 2005 in **Fig. 7**. In order to curb the influence of differences in the number of unique users, we normalized the number of the clusters, dividing by the total number of the threads. We call *cluster rate* this value. The result after the normalization is shown in **Fig. 8**.

Compared to the chart before the normalization (Fig. 7), the influence of the number of the unique users is suppressed, and it properly indicates the co-occurrence

of developers. The figure shows two characteristic phenomena at December 2004 and June 2005. Looking on the phenomena, the both cluster rates suddenly increase. Especially, the transition from April to June 2005 is very characteristic. By our investigation of the project, we found out a phenomenon that many developers left in the project from June 2005.

5. Conclusion and Future Work

We proposed an analysis method for measuring the activity of OSS project based on clustering of developers' co-occurrence feature. Then we measured the project activity affected by the co-occurrence of the developers through the experiment. The results of the experiment are concluded below.

- The statistical values of the cluster's elements cannot use as a measure of the project activity because it is affected by the number of emails.
- The number of the clusters cannot use as a measure of the project activity because it is affected by the number of unique users in the measuring period.
- The cluster rate suddenly increases at a point of the project change caused by any reasons.

In this result, we confirmed that the co-occurrence of developers measured by the cluster rate affects the project activity. However, we have not analyzed the whole period in the target project, and we have only analyzed one project. Our research needs to analyze more targets, and also needs to validate our method by statistical tests.

Acknowledgements This research is being conducted as a part of the EASE project in Comprehensive Development of e-Society Foundation Software program, and Stage Project, the Development of Next Generation IT Infrastructure, Japan supported by Ministry of Education, Culture, Sports, Science and Technology. This work is supported by Grant-in-aid for Scientific Research (B) 17300007, 2007, for Young Scientists (B), 17700111, 2007, and for Young Start-up, 18800024 by the Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

[1] J. Feller, B. Fitzgerald, S. Hissam, K. Lakhani, and W. Scacchi editors. Proc. of 5th Workshop on Open Source Software Engineering (WOSSE'05), 2005.

[2] J. Feller, B. Fitzgerald, W. Scacchi, A. Sillitti editors. Proc. of 3rd Intl. Conf. on Open Source Systems (OSS'07), 2007.

[3] H. Gall, M. Lanza, T. Zimmerman editors. Proc. of the Fourth International Workshop on Mining Software Repositories (MSR'07) , 2007

[4] A. Mockus, R. Fielding, and J. D. Herbsleb. Two Case Studies of Open Source Software Development: Apache and Mozilla, ACM Transactions on Software Engineering and Methodology (TOSEM), Vol.11, No. 3, pp. 309–346, 2002.

[5] D. German and A. Mockus, Automating the Measurement of Open Source Projects, In Proc. of 3rd Workshop on Open Source Software Engineering, pp. 63-67, 2003.

[6] G. Beenen, K. Ling, X. Wang, K. Chang, D. Frankowski, P. Resnick, and R. E. Kraut. Using Social Psychology to Motivate Contributions to Online Communities, In Proc. of 2004 ACM Conf. on Computer Supported Cooperative Work (CSCW'04), pp. 212–221, 2004.

[7] N. Hanakawa, K. Okura, A Project Management Support Tool Using Communication for Agile Software Development, In Proc. of 11th Asia-Pacific Software Engineering Conference, pp.316-323, 2004.

[8] Y. Yamauchi, M. Yokozawa, T. Shinohara, and T. Ishida, Collaboration with Lean Media: How Open-Source Software Succeeds, In Proc. of 2000 Conf. on Computer Supported Cooperative Work (CSCW'00), pp. 329-338, 2000.

[9] C. Bird, A. Gourley, P. Devanbu, M. Gertz, A. Swaminathan, Mining Email Social Networks, In Proc. of 2006 Intl. Workshop on Mining Software Repositories (MSR'06), pp.137-143, 2006.

[10] P. C. Rigby and A. E. Hassan, What Can OSS Mailing Lists Tell Us? A Preliminary Psychometric Text Analysis of the Apache Developer Mailing List, In Proc. of 2007 Intl. Workshop on Mining Software Repositories (MSR'07), pp.23-26, 2007.

[11] G. Salton, A. Wong, and C. S. Yang, A Vector Space Model for Automatic Indexing, Comm. of ACM, Vol. 18, No. 11, pp.613-620, 1975.

[12] Kaffe Project http://www.kaffe.org/