A Study on Growth Model of OSS Projects to estimate the stage of lifecycle

Yoshitaka Kuwata a*, and Hiroshi Miura b

a Muroran Institute of Technology, Japan  
b NTT DATA CORPORATION, Japan

Abstract

The products of Open Source Software (OSS) projects are widely used even in commercial mission-critical and high-availability systems. This is because both the quality of these software products is high enough for these applications and the support of software could fulfill the requirement. In general, when one wants to adopt OSS as a part of computer systems, it is required to examine the functional requirement (FR) for the OSS as well as nonfunctional requirement (NFR).

In the previous paper, we focused on NFR of OSS and proposed an evaluation method based on the maturity model of OSS community. Based on the model, we tried to evaluate four major OSS communities. For the evaluation, we used human knowledge of targeted OSS community. However it was not clear how to evaluate individual OSS project in OSS community.

In this paper, we focused on continuity of OSS project, as it is one of the most important factors for users to make a decision. In order to evaluate continuity, we proposed a growth model of OSS project, which is based on the size and activity of OSS Project. We evaluated the growth model using information retrieved from OSS communities from both OSS community sites and source code repositories.

Keywords: Evaluation of Software; Open Source Software Community; Source Code Repository
1. Introduction

Open Source Software (OSS)[16] becomes very popular and is widely used in commercial systems. In order to choose one of OSS required for ones’ purpose, it is very important to consider various conditions such as continuity, support, quality, and license. These conditions are called non-functional requirements (NFR). However, it is difficult to examine NFR of OSS only by examining source code.

In order to evaluate OSS based on their communities, we proposed a maturity model of OSS Community in Kuwata et. al. [10]. In the model, maturity of OSS communities are measured and used to grasp the status of projects and products. We applied the model to four major organizations and measured the maturity level. However, it is not clear the relationship between OSS community and OSS projects. For example, if an OSS community hosts several projects, the status of these projects should be different from each other.

In this paper, we propose an evaluation model of OSS project, and apply the model to projects for OSS communities.

2. Goal of this study

The goals of this study are as follows;
(1) Build evaluation models of the OSS community and OSS project.
(2) Based on the model, estimate non-functional requirement of OSS.
(3) Apply the model to actual OSS projects to evaluate the model.

3. Previous Research on analysis on OSS data

The development processes of most of OSS projects are open to public, and the record of development is available as on-line resources, such as archive of mailing-list, history of source code repository, issue tracking systems, wiki pages, and so forth. Many researches based on such on-line resources exists in software engineering research fields.

3.1. Source Code Repository

Recent years, one of notable movements is a group of researches on data from github[6], which is often called github mining. Github is very popular not just as a source code repository but as an on-line based collaborative software development environment. For example, if a developer wants to add functions to source code in a repository on github, he can “fork” the repository, modify the code and test in his local environment, and send “pull-request” of his code to the original repository. A series of comments and discussions will be carried out followed by decisions made by maintainers of the repository. The result can be “merged” or just “closed.” This style of collaboration becomes de facto standard among large-scale software projects. From the record of events on github, researchers can analyze the process of software development. Github Archive[8] provides all of events on github since 2011. One example of analysis is found in Jason et al. [17]. The paper analyzes a sample of extended discussion about pull-request and interviews the developer. Biazzini [1] focus on fork event and proposed a visualization method to show activities of projects.

On the other hand, Kalliamvakou [9] pointed out the promises and perils of mining github. For example, a repository is not necessary a project but used as code archive. The finding also includes that most projects on github are inactive, two third of projects are personal, and only a small part of projects uses pull-request. The median of the number of commit is 6, and many of projects do not update software often. As the usage of github is different according to the projects, it is difficult to estimate the state of project only by mining github.
3.2. OSS Community Site

OSS community sites are focus more on the aspects of human network of OSS. People use OSS community sites to compare OSS projects, find new OSS projects and people, and appeals their experience and carrier. Examples of OSS community site include advogato.org[12] and OpenHub[2].

In the case of OpenHub, which was called ‘Ohloh’ until 2014, the users create an account on OpenHub, claim their accounts on source code repository, and claim the role of OSS projects. Based on the information provided by users, OpenHub collects information from source code repositories and calculates various kinds of scores for users, projects, and organizations. An example is a score of contribution to OSS community. OpenHub provides information of OSS projects and OSS organizations as well as people involved.

Like github, OpenHub has functions to provide OSS data to external applications, which makes it possible for researchers to analyze OSS information on OpenHub.

Bruntink provides analysis tool and data cleaning model in [3]. Based on the analysis of OpenHub, Bruntink also proposed ‘code size’ and ‘growth of code’ as basic metrics of software in [4].

There is another attempt for the assessment of OSS projects, named RepOSS[13]. RepOSS tries building an online catalog of OSS based on information retrieved from various source code repositories. The site covers 340 projects but has no update since 2012.

4. Our Approach

In this paper, we try to estimate the status of OSS projects based on maturity model of OSS community. We newly introduce a project growth model, which is based on the stage of project lifecycle. In the model, a project is measured by its size and activity. When a project starts, for example, both size and activity are small (Born). If the project is on the track, the size and activity of the project are gradually increased (Childhood). When the project releases products, it is expected that the activity increase to a certain levels (Adolescence). The project keeps improving the products and the activity level. The size of the project can also be increased because of popularity of products. When the quality of the products reaches an enough level, the activity can be decreased while the size is kept high (Adulthood).

![Fig. 1 Overview of project growth model](image)

In order to measure the size and the activity of projects, there are several candidate indexes exist shown in Table 1. For the measurement of project size, ‘number of people involved’ and ‘code size’ are candidates. For the measurement of project activity, ‘number of discussion’, ‘number of code change’, and ‘size of code
change’ are candidates. We adopt ‘code size’ and ‘number of commits to repositories’, because these indexes are common among most repositories and organizations.

Table 1. Candidates of Index

<table>
<thead>
<tr>
<th>Index candidates</th>
<th>Project size</th>
<th>Project activity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Number of people involved (contributors)</td>
<td>Number of discussion</td>
</tr>
<tr>
<td></td>
<td>Number of users (download)</td>
<td>Number of code change (commits)</td>
</tr>
<tr>
<td></td>
<td>Code size (lines or bytes)</td>
<td>Size of code change (lines or bytes)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Number of pull-request (only in github)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Number of issues</td>
</tr>
</tbody>
</table>

It is expected that the growth model is common among the most OSS projects in an OSS community, because the management of the projects are the same. The patterns of past OSS projects might appear in new OSS projects in the same OSS community. In order to observe the difference of the growth models between different OSS communities, we are going to compare the growth models of OSS projects from different OSS communities.

We use both OSS community site and source code repositories to retrieve information of communities and their projects. OpenHub provides a list of projects supported by an organization. Based on the list of projects, we use source code repository to retrieve activity and code size.

5. Analysis of OSS Community Site and Code Repository

5.1. Overview of project analysis

We applied our growth model to projects hosted by two major OSS communities; Apache Software Foundation (ASF) [14] and OpenStack Foundation(OSF) [11]. These two communities are selected because we discussed in our previous paper. ASF is 20 years old and hosts about 200 projects. On the contrary, OSF is 5 years old and hosts 10 projects.

We also analyze docker [5] project to compare with projects in ASF and OSF.

Fig. 2 shows a scatter plot of projects in ASF, OSF, and docker project. A point in the figure represents a project. Round points are hosted by ASF, triangle points are hosted by OSF, and a square point is docker. X-axis is ‘the number of commits’ (in latest 12 month) to the project’s repository. Y-axis is the latest ‘lines of code’ in the repository. Note that X-axis and Y-axis are log-scaled.
Points for ASF projects are spreaded widely in the growth model; they differ four orders of magnitude in both the code size and the number of commits. On the other hand, most of OSF projects are in a small area. Comparing with ASF projects which is sized around 100K, OSF projects commits more actively.

Docker is located almost the same area with OSF project, which shows that docker project is also active. We are going to discuss about each communities in the following sections in details.

5.2. Case of Apache Software Foundation

Fig. 3 shows ASF projects with project name label. As the number of ASF project is large, major projects are only labeled. This graph shows projects are located different stages in lifecycle model. Projects located on the right-hand side are in active development, which include HBase, Cloud Stack, Spark, Hadoop, Cassanda, and Slang. On the other hands, several projects in left-hand side have large code size but no activity. An example of these projects is Apache Harmony, which has no activity since 2010.
Fig. 4 shows two types of typical projects from ASF. In this figure, the state of projects in the growth model is plotted by time sequences.

A scatter plot in left shows growth model of apache HTTP Server, which is the oldest project in ASF. Apache HTTP server is the most used HTTP server in the World. A point in this plot represents a state of the project in a certain month. For example, the development started 20 years ago, which is labeled 1996/7 in the bottom left corner. The development is still active in 2015/2 with more than 100 commits per month.

A scatter plot in right shows growth model of hadoop, which is one of the most active project in ASF now. The development of hadoop started in 2009/5. It is still actively developed. More than 100 changes are committed to the source code repository in a month.

As shown in Fig. 3, apache HTTP server and hadoop are located almost the same location in the code size and activity. From the time line analysis in the growth model, however, apache HTTP server is more stable and regarded as adulthood state, while hadoop has more commits than past, and is regarded as adolescence state.

5.3. Case of OpenStack Foundation

Fig. 5 shows OSF projects with name labels.
Unlike a graph for ASF, both X-axis and Y-axis are linear. Most of the OSF projects are located between 50K and 250K in code size, and between 1000 and 4000 commits per year.

Fig. 6 shows four projects in OSF as examples. In this figure, growth model is plotted by time sequences with labels. NovaCC (top-left) and Swift (top-right) are the oldest projects hosted by OSF. These two projects are launched in 2010/7 and increase the size as time goes by. Although the size of these projects is different, the shape in growth model looks similar. We regards these projects are in adulthood stage.

On the contrary, Quantum (bottom-left) and Horizon (bottom-right) increase their activity in 2014. We regard these projects are in childhood or adolescence stage.

5.4. Case of Docker Project

Docker project is selected for the analysis because the project develops the same category of software with OSF projects but the project is younger than OSF projects. Fig. 7 shows docker project in a time line of growth model. Docker started in 2013/1 and has more than 500 commits per month in 2015/1. The code size and activity is similar to Quantum project in OFS. The shape in the growth model is similar to apache hadoop project. As the project is very active and size of code keeps growing, the project is in childhood to adolescence stage.
6. Issues and Evaluation

6.1. Decision make with growth model

We examined several projects from ASF and OSF in details. In general, projects in adulthood stage are stable. The products released in this stage of projects are suitable to apply for commercial systems, as the quality of products is good and the products are maintained well.

The products released by projects in adolescence stage are also good for commercial systems. They have a small risk that the products are updated often and the support of older products ends.

The products produced by projects in childhood stages should be used carefully. As the growth of the project is very high in this stage, the specifications of the products can change in a short period of time. The defects can exist in the products of this stage. Thus, only advanced systems, which require cutting-edge technology, should use these products with their own risks.

The products released by obsolete projects should never be used in new projects. If the products are used in commercial systems, they should be replaced as soon as possible. In general, products should be upgrade before the project becomes obsolete.

6.2. Limitations of OSS Community Site

As OpenHub provides information obtained by people, there are limitations in coverage and accuracy. For example, there is no information exist for Free Software Foundation (FSF)[15]. Information of projects hosted by FSF exists in OpenHub. In order to apply growth model to FSF projects, we need to obtain a list of projects of FSF from other information source.

OpenHub also provides statistics obtained from source code repository. As the retrieval of repository is batch based, the information can be a few months late. To obtain up-to-date information, retrieval engines should be used for analysis instead of OpenHub. One example of original retrieval engine is found in Jesus et. al.[7].
6.3. Limitations of the growth model

The growth model assumes that projects start from scratch. It is also assumed that source code repositories are used for development of software. If the growth of project is not fit in this model, the projects need to be carefully examined for these assumptions.

We found that the projects grow non-continuously in the following cases.

1. Project split into several projects in the middle of development
2. Project started as non OSS project and switched to OSS
3. Project started as OSS project and switched to non OSS
4. Project is abandoned

7. Conclusions and future works

We proposed a growth model of OSS projects and apply the model to two major OSS communities. We found that we can use the model to understand the status of OSS projects with their OSS community. We also found that not all of projects follow the model. We are going to apply the model to other OSS communities to evaluate the applicability of the mode.

Established OSS projects, such as Linux, has also concrete organization and has series of technical conference for designing, archiving consensus of design, targeting current issues, and et al. It may also have a series of user symposium, attracting ICT users, providing training opportunities, sharing use cases and best practices etc. These activities reinforce organizations because of sharing goal, motivating developers and its sponsors. Examining articles and session agendas may be another data set for analyzing OSS maturity level.

References

