# A Novel Open Source Software Ecosystem: From a Graphic Point of View and Its Application

Chenxi Song, Tao Wang, Gang Yin, Xunhui Zhang, Cheng Yang
National Laboratory for Parallel and Distributed Processing
College of Computer, National University of Defense Technology
Changsha, China
songchenxi92@163.com, { taowang2005, yingang }@nudt.edu.cn, zhangxunhui9368@163.com, delpiero710@126.com

***Abstract*—With the rapid development of open source software, various elements such as OSS, developers, users and online posts, across different communities and their interactions constitute a novel software ecosystem. Most of the current researches about software ecosystems care the connections between software, and few of them consider the relationship across communities from an overall perspective, and fail to cover the users and their activities which should be an indispensable part of the OSS ecosystem. This paper model the OSS ecosystem as a graph, which combines different types of OSS communities as a whole. Based on this graph model, we analyze the characteristics of ecosystem, like the evolution, competition and symbiosis. In addition, we build a recommendation system as well, and the experiment results suggest the validation of our approach.**

***Keyword; open source ecosystem; graph model; recommendation algorithm***

## I. Introduction

In recent years, there has been a surge of interest in open source software (OSS, for brief) development and most of them focus on single community. As far as we know, there are a large number of communities related to open source software, such as GitHub, StackOverflow, OpenHub, OSChina, etc. Some of them focus on project developing and software hosting, like Github, and others focus on knowledge sharing, like StackOverflow. Therefore, we divide these communities into two categories, software development community that can provide abundant repository data and development process reports, and knowledge sharing community that contains a large amount of discussion about OSS. In this paper, we connect different communities together, constitute an OSS ecosystem, extract various types of OSS related data and organize them into a graph model.

Since Messerschmitt and Szyperski in [1] propose the conception of *software ecosystem* in 2003, the researches on software ecosystems have been around for more than a decade. Surveys [2] [3] about different definitions and research emphasis of software ecosystem show that there has no consistent definition through more than 20 research works. These works all contain a collection of software and some kinds of relationships either symbiosis, dependency [5], common evolution [4], business or technical. To the best of our knowledge, few of them combine different communities together taking advantage of the feedback from users.

In this paper, we propose a novel definition of open source software ecosystem, which combines different elements from a variety of open source communities for the first time. Firstly, we construct a graph model to describe the software ecosystem, in which each element is seen as a vertex, relation as edge. Secondly, we analyze characteristics of the ecosystem such as the symbiosis, competition between software. Afterwards, from the graphic view, we build a recommendation system that recommends related software to users. Compared with previous work, our ecosystem covers a variety of open source communities. Combining different communities together, we can analyze the development process of OSS and relations between them like cooperation and competition, through a global perspective.

This paper is organized as follows: section II describes the definition and graph model of OSS ecosystem as well as the recommendation approach; section III introduces the experiments of analyzing characteristics of the ecosystem; experiment results and the evaluation of recommendation are in section IV, followed by a discussion of threats to validity; we conclude with a discussion of our contribution in section V.

## II. Oss Ecosystem

This section will introduce the definition of our open source software ecosystem. Since different elements in the ecosystem interconnect with each other, we map the ecosystem to a graph model. Based on the graph, we design a recommendation system, which is described in this section as well.

### A. Background and Definition

In ecology, an ecosystem is a community of living organisms in conjunction with the nonliving components of their environment (things like air, water and mineral soil), interacting as a system [10]. In open source area, there exists elements of software development community like software, developers and elements of knowledge sharing community like online posts, users, followers etc. The associations between elements constitute an OSS ecosystem.

We define the OSS ecosystem as **a set of open source software along with their contributors, followers, users, and the social message generated by the development activities**

DOI reference number: 10.18293/SEKE2016-123

**and users feedback.** The ecosystem runs as Fig.1, in which the direction of arrows expresses the flow of products.

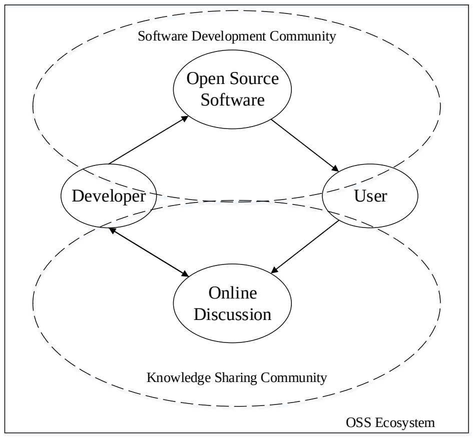


Figure 1. Actors in the OSS ecosystem

In ecology, ecosystem contains producers, consumers and the ecosystem dynamics, which are presented in the interactions between different species [4]. Besides the food chain of natural ecosystem, there also exist relationships like symbiosis, competition, etc. In our software ecosystem, these characters and relationships have corresponding expressions.

The producers of our software ecosystem are, of course, the developers and the software itself. Consumers are users. However, unlike the consumers in natural ecosystem, the software consumers also give feedback to producers, share their use experience, report bugs and make demands through online posts and other messages. The food chain is the flow of OSS. Symbiosis and competition mainly exists between OSSs, and we will introduce them in section IV.

## *B. Graph Model for OSS Ecosystem*

The ecosystem contains developers, OSS, users and online posts in general. Since tag is very common in communities, we also take it into consideration. Based on the interactions between different elements, we map the ecosystem to a graph model, represented as G = (V, E). In the directed graph G, each vertex u ∈ V represents an element of open source ecosystem and the set of edges E contains all relationships that are present between these two distinct sets of vertexes. Both vertexes and edges can have labels and properties. The graph model is shown in Fig. 2.

Five types of vertexes exist in the graph model, namely: *Software*, *Tag*, *Community*, *Post* and *Person*. *Software* is the essential element, *Tag* is label of software or posts in communities for category or subject indexing, and *Community* identifies the source of software or posts.

The seven kinds of edges are based on relationships and interactions between elements. The direction and property of edges can be seen from Fig.2. Edges are the important bases for the analyzing of OSS ecosystem. For instance, the *DISCUSS_ABOUT* edge is defined as

$$e_{DISCUSS\_ABOUT} = \{<u, v>|u \in V_{Post}, v \in V_{Software}\},$$

which means post *u* is a discussion of software *v*. Particularly worth mentioning is that we have a specific algorithm to bridge open source software and online posts.

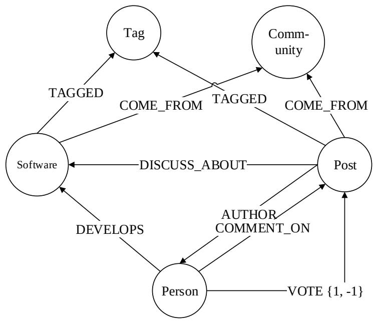
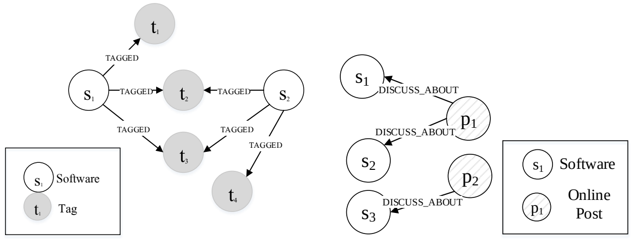


Figure 2. The graph model of OSS ecosystem

## *C. Recommendtaion Approaches*

A primary application of the OSS ecosystem is a recommendation system that can recommend relevant software for a specific software that user required. The general approach is that once a user require for a specific software $s_0$, we first calculate the relevancy between $s_0$ and other software, and recommend the most relevant ones.

The relevancy between two OSSs is calculated based on two aspects, namely: tags and posts, denoted as $r_t$, $r_p$. Fig.3 is a diagram of the two relevancies.



(a) Examples of tag relevancy (b) The co-occurrence of software

Figure 3. Relevancy calculation

*Tag relevancy*: After analyzing the tag usage of OSS ecosystem, we find that tags indicate the technology, language or feature of the software. So containing same tags shows some relevance of two OSSs. However, some tags like "java" is so common that only calculating the intersection may lead to incorrect OSSs. So we draw lessons from TF-IDF and calculate a word frequency weight for each tag.

For the tag $t_i$ of OSS $s_j$,

$$\mathrm{tf_{ij}} = \frac{n_{ij}}{\sum_k n_{kj}} = \frac{1}{\sum_k n_{kj}} \tag{1}$$

$$\mathrm{idf_{ij}} = \log\frac{|S|}{|\{j : t_i \in s_j\}|} \tag{2}$$

$$\mathrm{w_{ij}} = \mathrm{tf_{ij}} \times \mathrm{idf_{ij}} \tag{3}$$

In TF-IDF model, $n_{ij}$ of Eq. (1) means the number of appearances of tag $t_i$ in software $s_j$, which is 1 in our scenario. And $\sum_k n_{kj}$ represents the total tag number of $s_j$. In Eq. (2), $|S|$

is the total number of OSSs, and $|\{j: t_i \in s_j\}|$ represents the number of OSSs that contain tag $t_i$. The TF-IDF value is used as the weight of each tag and transforms the set of tags into vectors. Then *cosine similarity* is used to calculate the tag relevancy between two software.

*Post relevancy*: As we mentioned before, users may post in communities to discuss one or more open source software, which is indicated by the *DISCUSS_ABOUT* relationship in the graph model. OSSs with related or similar function are often discussed together, we call this phenomenon the co-occurrence of software. After analyzing the different OSSs discussed in the same post, we find that these OSSs often have related functions or same user groups, the statistics is shown in section V. So we treat the co-occurrence phenomenon as an important basis of recommendation.

For a particular software, we first gather all OSSs that are discussed with $s_0$ in same posts and their co-occurrence frequency. The relevancy of posts defines as the normalized frequency, whose range is between 0 and 1.

*Recommendation Model:* In order to synthetically consider the influence of two factors, we assign each relevancy a weight value, α and β. And calculate the final recommendation score as:

$$R = \alpha * r_t + \beta * r_p \quad (4)$$

Find the top-k list of the final relevancy score *R* with regard to the specific software as recommendation result.

## III. Experiments Design

Symbiosis, competition and evolution are significant features of ecosystem in ecology, which have corresponding expressions in our OSS ecosystem. In this section, we present our research questions and experiment settings of analyzing ecosystem characteristics.

### *A. Research Questions*

OSS ecosystem connects different communities together, which gives us an opportunity to analyze the characteristics of open source environment in multiple views. The bridge between open source software and online posts remarkably leverage the crowd wisdom to support the development of OSS. In this section, we would like to analyze the OSS ecosystem in three aspects.

#### *1) The symbiosis and competition of open soruce software*

In software ecosystem, different software with same or similar functions may have competitive relationships, like Ubuntu and Fedora. Collaborations exist as well, such as MySQL and the MySQLWorkbench, which is called symbiosis in natural ecosystem. We located the two relationships in software ecosystem through co-occurrence phenomenon.

#### *2) The evolution of open source software*

Species in ecosystem evolve over time to adapt to the changing environment and reproduce, so does open source software in our ecosystem. Discussions from usrs is an important reflection of OSS evolution. We study the change of discussion number and discuss the evolution of OSS.

#### *3) A recommendation application based on graph model*

Based on the graph model and characteristics mentioned before, we accomplished a recommendation application to propose related software when users require for a specific OSS.

### *B. Dataset and Experiment Settings*

In order to collect enough OSS information for establishing an integrated OSS ecosystem, we use a web crawler to gather realistic sets of OSS data from dozens of communities and get a total of more than 230,000 software and 5,100,000 online posts. Since the data quality in these websites varies greatly, we select the valuable and filter out the noises. At last we obtain about 11,000 software and 100,000 online posts and its users, tags as well as information of communities for building the OSS ecosystem and conducting the experiments.

## IV. Experiment Results

Our ecosystem associates online posts with open source software, which gives us a chance to seek the overview of the ecosystem. In this section, we analyze the characteristics in OSS ecosystem and evaluate the recommendation system. Meanwhile we describe some threats to validity.

### *A. Ecosystem Characteristics*

Limited by the data source, we cannot build the relations between OSSs (Fig.2). However, we can still catch a glimpse of the relationships of OSS with the aid of discussion posts.

#### *1) The symbiosis and competitive relationships*

When online posts discussing two or more software at the same time, these software must have some connections. For example, app developers may share their experience in using *SQLite* while talking about the feature of *Android*. These two software cooperate in developing Android apps. In this occasion, we think the co-occurrence software have symbiosis relationship. Also, users may compare several similar software, and these software are competitors.

We collect 500 sets of software that has co-occurrence relationship and find that 87.2% of them are the symbiosis relationship. Only 9% are competitors and others 3.8%.

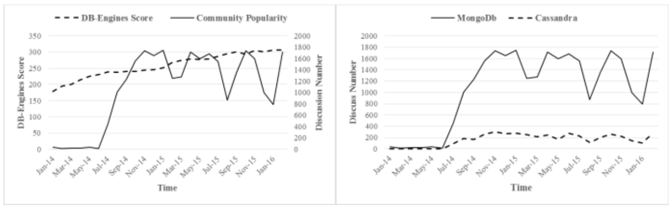
#### *2) The evolution of open source software*

*Darwin*, the famous naturalist and geologist, proposed “Survival of the fittest” evolution theory. It also fit for software ecosystem. Similar to the method in 1), online posts are the basis element of analyzing software evolution.

Firstly, a popular software may be discussed more than others. Take *MongoDB*, a popular NoSQL database, as an example. We gather statistics of its discussion number of each month and compare it with the scores given by DBEngines[1], a famous DBMS ranking website (Fig. 4(a)). The fluctuation of discussion number may be caused by some events, for example, a new version released. An apparent ascends in April 2015 may due to the release of 3.0. Each time a new version released users enjoy discussing the new features. As time goes by, this enthusiasm of the discussion may gradually reduce. Hence, the discussion number can represent the development of software to some extent.

[1] http://db-engines.com/en/ranking_trend/system/MongoDB/

Also, the evolution of software may influence the development of competitors. Fig.4 (b) shows the growing trend of *MongoDB* and *Cassandra*. As competitors, *MongoDB* gets the support of more users and has higher rank in DBEngines.



(a) The community popularity and DB-Engines score of mongoDB (b) The comparison of discussion number

Figure 4. The evolution of software

## B. *Recommendation Evaluation*

In order to evaluate the performance of recommendation system objectively, we compare our output with the recommendation of OSChina. OSChina[2] is the biggest open source community of China, which is one of our data sources. Take *MySQL* as an input example, the recommendation results of two algorithms is shown in Table I.

TABLE I. MYSQL RECOMMENDATION RESULTS

| **Our approach** | **OSChina** |
|---|---|
| - PDO<br>- phpMyAdmin<br>- Hibernate<br>- WordPress<br>- MySQL Connector/J | - MySQL MTOP<br>- MySQL Installer<br>- Google MySQL<br>- MySQL Syncer<br>- MySQL Utilties |

As Table I shows, OSChina recommend MySQL tools which used to improve the performance in data storage, management and synchronization. However, it seems that these approaches pay too much attention to the software itself. Our approach, by contrast, provides a more practical choice. *PDO* and *PhpMyAdmin* are widely used for accessing and managing MySQL dataset in *PHP* software developing. *Hibernate* and *MySQL Connector/J* are the encapsulation for JDBC in Java applications development. *WordPress* is web software to create bolgs and apps, which use MySQL to do data storage.

The options that our recommendation system give, are popular developing tools cooperating with MySQL. When a user searches for MySQL, he may need these interfaces or management tools to set up the whole system. The advantage of this result is that when a user requires a specific software, some popular and related tools will be recommended. It helps users to notice the hot technologies of related software and there is high possibility that they are using or intend to use one of them.

## C. *Threats to Validity*

Firstly, the data in knowledge sharing communities may have some noise. The online posts or technique news don't have to be only related to open source software. It may also talk about some latest technology or some business software. We have designed a particular program to filter these data and minimize the noise. Secondly, the relationship between open source software and online post is established by a match algorithm which conduct textual analyze to find the software that a post discuss about. In our testing, the bridging algorithm has a high accuracy of about 90%, and we filter some error before using to minimize the impact of errors. But it can still bring adverse effect to the ecosystem and its analysis. Also due to our limited datasets and parameters used in our approach, the evaluation is not comprehensive enough.

## V. CONCLUSION

Software ecosystem has been a hotspot in software engineering area. In this paper, we collect millions of open source software, online posts and user information from dozens of open source communities. The association and interaction from the variety of data constitute an OSS ecosystem. We study the software evolution, as well as the symbiosis and competition between software, which are the important notions in ecology ecosystem. Viewing the elements in the ecosystem as vertexes, the relations as edges, we map the software ecosystem to a graph model, which helps us to get a full appreciation of the distribution and characteristics of ecosystem. Also, based on the analysis and the graphic view, we propose a novel recommendation approach and evaluate the result critically.

## ACKNOWLEDGMENT

This research is supported by the National Natural Science Foundation of China (Grant No.61432020, 61472430 and 61502512).

## REFERENCES

[1] D. Messerschmitt and C. Szyperski, *Software Ecosystem: Understanding an Indispensable Technology and Industry*. MIT Press, 2003.

[2] K. Manikas and K. Hansen, "Software ecosystems – A systematic literature review", *Journal of Systems and Software*, vol. 86, no. 5, pp. 1294-1306, 2013.

[3] A. Serebrenik and T. Mens, "Challenges in Software Ecosystems Research", in *European Conference on Software Architecture*, 2015, p. 40.

[4] T. Mens, A. Serebrenik and A. Cleve, *Evolving Software Systems*.

[5] L. Šubelj and M. Bajec, "Community structure of complex software systems: Analysis and applications", *Physica A: Statistical Mechanics and its Applications*, vol. 390, no. 16, pp. 2968-2975, 2011.

[6] G. Yin, T. Wang and H. Wang, "OSSEAN: Mining Crowd Wisdom in Open Source Communities", in *Service-Oriented System Engineering*, 2015, pp. 367-371.

[7] S. Bajracharya, J. Ossher and C. Lopes, "Sourcerer: An internet-scale software repository", in *ICSE Workshop on Search-Driven Development-Users, Infrastructure, Tools and Evaluation*, 2009, pp. 1-4.

[8] H. Wang, T. Wang, G. Yin and C. Yang, "Linking Issue Tracker with Q&A Sites for Knowledge Sharing across Communities", *IEEE Transactions on Services Computing*, pp. 1-1, 2015.

[9] Y. Yu, H. Wang, G. Yin and T. Wang, "Reviewer recommendation for pull-requests in GitHub: What can we learn from code review and bug assignment?", *Information and Software Technology*, vol. 74, pp. 204-218, 2016.

[10] "Ecosystem", Wikipedia. [Online]. Available: https://en.wikipedia.org/wiki/Ecosystem.

[2] http://www.oschina.net/