Towards Understanding Developers' Collaborative Behavior in Open Source Software Ecosystems

Yaxin Liu[1], Peng He[1, 2*], Gaoyan Wu[1], Yilu Li[1]

[1] School of Computer Science and Information Engineering, Hubei University, 430062, Wuhan, China.
[2] Hubei Province Eng. Technology Research Center for Education Info., Hubei University, 430062, Wuhan, China.

* Corresponding author. Tel.: 15623729083; email: penghe@hubu.edu.cn

Manuscript submitted March 20, 2017; accepted May 10, 2017.
doi: 10.17706/jsw.12.6.393-405

Abstract: Understanding developers' collaborative behavior is an essential step to facilitate the needs of collaborative development activities in open-source software (OSS) ecosystem. In this paper, we conducted an empirical study to gain insight into how existing developers collaborate together and what factors affect their collaborative behavior from the perspectives of collective interaction, and attempt to explore the typical interactive modes for the corresponding types of new collaborations. The results show that different existing developers tend to prefer different collaboration patterns, and short topological distance has very limited effect on their first collaboration. By contrast, development experience seems to be important factor that affect their potential collaboration. The results also indicate that different collaboration patterns support different interactive modes, and are related to the popularity of projects. The findings are of value for existing developers to maintain sufficient collaboration awareness, so as to improve the stability and sustainability of OSS ecosystem.

Key words: OSS ecosystem, developer collaboration network, behavior analysis, collaboration pattern.

1. Introduction

A decade ago, Ye and Kishida [1] argued that an OSS (Open Source Software) project is unlikely to be successful unless it is accompanied by a developer community. Today, single OSS is no longer alone but rather an individual (a system) correlated with others through the sharing of environment and resource in a software ecosystem [2]. OSS ecosystem has come to prominence in global development settings where distributed cross-cultural developers collaborate with each other, facilitating those cooperative development activities that occur in different places and at different times.

In general, an open source community (e.g., Sourceforge), comprising of a set of software projects which are developed and evolve together in the same environment, can be viewed as an OSS ecosystem. From the viewpoint of cooperative activity, it can be modelled as a self-organizing collaborative social network, including thousands of world-wide developers who prefer to work voluntarily for their enjoyment and pride of being part of a successful virtual community [3]. Although these developers often come from different geographic locations, within the ecosystem they can work closely as a team to share information, knowledge and achievements that incorporate their individual contributions and creativity.

Numerous researchers pay great attention to understanding and explaining a wide variety of phenomena occurred in OSS ecosystem, such as the determinants for OSS success [4], [5], developers' motivations as a volunteer [6], cooperative mechanism and awareness [7], the evolution of developer communities [8], and so forth. However, one of the most important issues on collective collaboration—the interactive behaviour among developers within their ecosystem—remains unsolved. Indeed, human behaviour is one of the significant issues in science, which has a history of about one century since the time of Watson [9].

To the best of our knowledge, few of researchers have made some attempts to address the issue (which is the developers' collaborative behaviour in OSS). In literature [1], the authors found that newcomers moved from the periphery to the core in the onion-model of the OSS community analyzed through continuous learning with other members. Bird *et al.* [10] analyzed the transition from non-developers to full developers, and found that both technical skill and social reputation are important impact factors. Unfortunately, the prior studies focus mainly on the behaviour of newcomers in a single OSS community; on the other hand, the transition of only a few newcomers can be successful during this process [7]. That is to say, the interactions among existing developers may be more valuable to understand developers' collaborative behaviour in an OSS ecosystem. Nevertheless, In order to gain an insight into the above-mentioned issues, the goal of this paper is to investigate the collaborative behaviour of existing developers in an OSS ecosystem. We have defined the following three research questions:

RQ1: What characteristics of the collaboration among existing developers are shown in OSS ecosystem?

RQ2: What factors affect the collaborations among existing developers in OSS ecosystem?

RQ3: What are the typical interactive modes among existing developers in our context?

The contribution of our work is twofold:

1) In contrast to most of prior studies, we investigate the potential (first and recurring) collaboration among existing developers from the perspectives of both collective interaction and historical experience, which is an important step towards understanding the stability and sustainability of an OSS ecosystem.
2) Our findings would provide new insight into the proper peer recommendation for a target developer and the sound coordination with other members in an OSS ecosystem.

The rest of this paper is organized as follows. Section 2 is a review of related work. In Section 3, we introduce some preliminary knowledge based on the framework of an approach to behavior analysis with several quantitative and qualitative measures involved in our experiments. In Section 4, we present an experimental study to illustrate the collaboration in an OSS ecosystem, and discuss our findings. Section 5 is the threats to validity. Finally, we conclude our work and present an agenda for future work in Section 6.

2. Related Work

Most prior studies focus mainly on understanding the motivations, participation and performance of developers, as well as how different motivations affect developer's participation and performance in OSS projects. Ye and Kishida [1] argued that learning is one of the driving forces that motivate developers to get involved in OSS projects because it provides intrinsic satisfaction for developers, and the role transformation of developers in a community by learning offers extrinsic motivation. Roberts et al. [6] found that developers' motivations are not always independent, but rather are related in complex ways; moreover, different motivations have an impact on their participation in different ways. Beecham et al. [11] performed a literature review about motivation in software engineering, and they reported some motivational factors, e.g., affinity with the task, good management, involvement/ team work, and so on.

On the other hand, there have been several researchers who investigated the interactions between newcomers and existing developers, including what happens when a newcomer joins a software project, and which factors may affect his/her growth. Bird et al. [10] studied the phenomenon of "immigration" in OSS projects and found that prior actions have a very strong effect on becoming a developer. Krogh et al. [12] investigated the joining script of newcomers and discovered that a significant period of observation was needed before someone felt that he/she could contribute to a technical discussion. Fronza et al. [13] studied how newcomers joined agile projects, and they found that pair-programming was used to initiate newcomers into a project. Herraiz et al. [14] found that volunteers tended to follow a step-by-step joining process, while firm sponsored developers usually experienced a "sudden" integration.

Dagenais *et al*. [15] concluded several primary factors that influence the integration process of newcomers, e.g., the need for early practice. Ducheneaut [16] pointed out that prior technical activity and social standing in the community are strong factors in achieving developer's status. Hahn et al. [17] examined how OSS project developer teams form and developers' joining behavior, and they tested the impact of previous collaborative ties on software team formation.

Besides, a few researchers tried to investigate this issue from the perspective of social networks. Toral *et al*. [18] applied social network analysis techniques to identifying "brokers", who play a middle-man role among different developer communities. Crowston *et al*. [19] identified leaders in terms of the highest level of contribution he/she made, and they assessed the degree of leadership by measuring the centralization of communication. Hossain *et al.* [20] explored the coordination performance of OSS projects on the SourceForge.net (SF), and they also investigated the correlations between both density and centrality and coordination performance of defect management systems. Datta *et al*. [21], [22] studied the evolution of developer collaboration for an agile project on the Jazz platform. They concluded that developers were more likely to collaborate as iterations proceed towards a release, because the parameters of collaboration change rapidly in earlier iterations than later ones.

Huang *et al*. [23] proposed a developer-module network to describe the interactions among developers, and they used a Legitimate Peripheral Participation (LPP) model to divide developers into core and peripheral teams, based on the evolutionary process of learning behavior. Allaho and Lee [24] analyzed the connectivity structure of developer networks and investigated the effect of different social ties on developers' overall productivity. The results showed that there existed significant interactions between experts and newbies, and that there were positive correlations between in/out-degree and developers' contributions in OSS projects.

It has recently been suggested that software ecosystem is an effective way to construct large software system/platform by collecting different components/software developed by the same community [25]. In these setting, Hanssen concluded that opening up both the technical and organizational constraints are key enablers to achieve an increase in collaboration across organizational borders and resource sharing [26]. Gamalielsson and Lundell [27] considered a long-term study of the sustainability of OSS community. They analyzed how specific OSS ecosystem were affected by a fork and found that there are no significant signs of stagnation in the target project after the fork. Posnett *et al.* [28] analogized the developer-aritifact contribution network to a predator-prey food web, and measured the focus and ownership from both the developer and artifact perspectives. The authors found that more focused or specialized developers often introduce fewer defects than those defocused ones.

In this paper, we will concentrate on those existing developers in terms of software ecosystem. Therefore, our motivation is to understand the collaborative behavior of existing developers, rather than those newcomers (or all developers). Although it has been well recognized that understanding the development of OSS communities has become a meaningful topic, few researchers have addressed the issue as we will do from an ecosystem perspective.

3. Preliminaries

3.1. OSS Ecosystem

With the popularity of the Internet, more and more developers geographically distributed in the world get involved in OSS projects, and then different communities where projects are deployed freely and teams are organized loosely were formed, also named as software ecosystem. In this paper, we focus on the OSS ecosystem composed of those developers who contribute to the projects hosted on the SourceForge.net. It has been recognized that SF is one of the most representative OSS movements, in part because of its popularity and the large number of registered developers and projects [29].

There are more than 300,000 registered OSS projects and more than 2 million developers or users in the SF. So far, the numbers of projects and developers are still ever-increasing. To the best of our knowledge, there are over 100 tables in the data dump of the SF. Considering that the goal of our study is to analyze developers' collaborative behavior, three original attributes were employed in this paper, including 9 tables: developers, projects, developer_projects, project_programming_language, project_intended_audience, and project_license. We collected the historical data related to the nine tables provided by the Flossmole.org [30]. Specifically, we only used the data from April 2007 to April 2008 because of the sound integrity of such data. In order to explore the impact of registration time difference between two developers on their collaboration, we also crawled join date for each developer. The details of mentioned attributes involved in our experiments are listed in Table 1. The last column represents the number of elements of each attribute. For example, the Intended_Audience attribute refers to 16 elements: Healthcare, Education, etc.

In order to reduce the efforts of data processing, we selected those projects with the topic "software development" and complete attributes in the six tables as our objects. In addition, the projects with only one developer were also filtered out. Note that, "software development" is the largest branch of topic attribute in the SF ecosystem, and the data is collected once every two months.

3.2. Existing Developer Network: EDN

Note that, the developers discussed here are limited to the core developers who have made significant contributions to the development and evolution of an OSS project, rather than all members in it. Therefore, one of our underlying assumptions is that a relationship between two developers occurs when they participate in the same project. Then the collaboration among developers can be characterized by a developer network, where the nodes represent developers and the (undirected) edges correspond to their collaborative relationships. The edge weight is simply equal to the number of projects where two developers in question have worked together.

Fig. 1 shows a simple example of developer network, where the thick solid lines and the dotted lines on the right represent the repeated and the initial collaboration between existing developers, respectively. However, since the goal of this paper is to analyze the collaborative behaviour of existing developers in an OSS ecosystem, we did not consider the addition of newcomers and the cooperation among them.

The brief information of developer networks is listed in Table 2. |*| represents the number of elements in the set *; the subscript s and e denote the initial (April 2007, abbreviated as 0704) and the latest time interval (April 2008, abbreviated as 0804), respectively.

Table 1. Description of Attributes Mentioned

Attribute	Description	# of Elements
Intended_Audience (IA)	The target objects that an OSS project serves determine the range of developers.	16 (e.g., Healthcare, Education)
License	The license used by a project can influence the amount of support it gets, as it affects the interests of developers.	20 (e.g., BDS, GPL)
Programming_Language (PL)	The development language used affects the implementation of an OSS project.	22 (e.g., JAVA, C++, PHP)

Table 2. Statistics of Developer Collaboration Networks Analyzed

$\|Node_s\|$	$\|Edge_s\|$	$\|Node_e\|$	$\|Edge_e\|$	$\|Interval\|$	$\|Month\|$
10727	70096	11826	81424	6	12

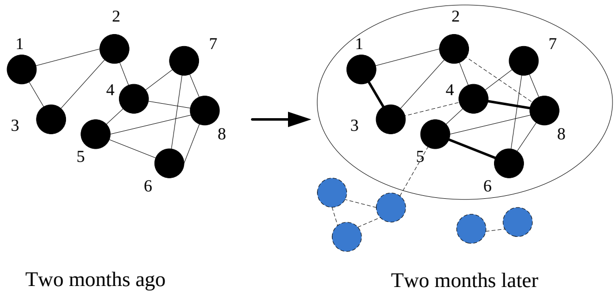


Fig. 1. An example of existing developer network. (The black nodes represent existing developers and the blue nodes represent new developers).

3.3. Behavior Analysis

There are four types of typical collaborations in a developer network (see Fig. 2). That is, a potential collaboration may be established (1) between two newcomers (e.g., the edges between the blue nodes), (2) between a newcomer and an existing developer (e.g., the edge between node 5 and the blue one), (3) between two existing developers who did not work together before (e.g., the edge between node 3 and node 4), and (4) between two existing developers who have collaborated with each other at least one time (e.g., the edge between node 1 and node 3). It is worthwhile to note that only the last two patterns, also labeled as o1o and o2o, will be mainly discussed in the following sections. The analysis of other two patterns has been studied in a previous work [31],

As we know, "friends of friends" is often used to recommend interested followees for users in social network. Therefore, the studying whether the short topological distance between two existing developers affects collaborative behavior of developers is an interesting work, in particular for the o1o collaboration. Thus, in a developer network, we use the number of hops between two developers to measure the effect of topological distance on o1o collaboration.

Besides, the similar or complementary development experience is also likely to cause the possibility of collaborations among those developers. Using the same approach proposed in our prior work [32], we compute each existing developer's expertise through quantifying a set of project attributes listed in Table 1, represented as Expertise=(AT_{ia}, AT_{l}, AT_{pl}) . Specifically, we used a vector of element AT_{i} weights for each developer to represent a specific attribute. The element weight w was computed based on the TF-IDF method [33]. In our context, tf and idf are defined as follows:

$$tf_{ed} = \log(f_{ed} + 1)，\ idf_e = \log\left(\frac{\#devs}{\#dev_e}\right). \tag{1}$$

where f_{ed} refers to the number of occurrences of an element e for developer d , whereas $\#devs$ and $\#dev_e$ represent the number of developers and the number of developers who has the element e of a specific attribute, respectively. Then, we calculated each element weight w for developer d in terms of the following equation:

$$w_{e\in d} = tf_{ed} \times idf_e = \log(f_{ed} + 1) \times \log\left(\frac{\#devs}{\#dev_e}\right). \tag{2}$$

Finally, the similarity between two existing developers was calculated in terms of the cosine similarity between their corresponding attribute vectors. To simplify the computation, all coefficients i is set to 1.

$$Sim(A,B)=\sum_{i=1}\alpha_i\cos(AT_{Ai},AT_{Bi})$$
$$=\sum_{i=1}\alpha_i\frac{\sum(w_{Aij}\cdot w_{Bij})}{\sqrt{\sum_{j\in Ai}(w_{Aij})^2}\sqrt{\sum_{j\in Bi}(w_{Bij})^2}}. \tag{3}$$

4. Experiment Results

RQ1: What characteristics of the collaborations among existing developers are shown in OSS ecosystem?

On the basic of SF ecosystem under discussion, we firstly analyzed the evolution of the structure of existing developer networks and found some interesting results. For instance, the average shortest path length shows a trend of fluctuate decrease (see Fig. 2(a)), which indicates that the connectivity of existing developers becomes higher because of a large number of new collaborative relationships; the stable trend of high clustering coefficient, shown in Fig. 2(b), implies that existing developer's collaborators are not willing to work with each other when establishing new cooperation, and it is not in conformity with the common sense of "friends of friends" on general social networks; Fig. 2c) shows the significant unequal distribution of node degree, since the Gini coefficient is greater than 0.6, suggesting that there exist a small number of developers who collaborated with many other developers in the SF ecosystem.

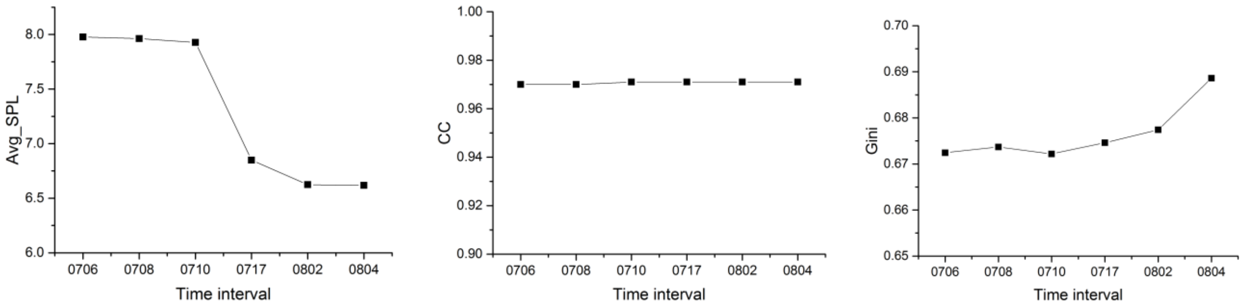
Additionally, according to the attributions listed in Table 1, we also attempted to investigate that whether the collaboration under discussion is more likely to occur in a certain kind of project. In the light of the results presented by Fig. 3, the answer is obviously positive because of the significantly and consistently skew distribution of the ratio of new collaborations. That is to say, the probability of new collaboration in different types of projects is varied. We used the dotted ellipse to mark the element with highest ratio, and gave out its label. For instance, more than 60% of new collaborations were occurred in those projects that serve the developer, and about 40% collaborations were occurred in Java projects.

The above analysis merely shows us several rough clues of the collaborations among existing developers. We further recorded the number of o1o and o2o collaborations in each time interval. Fig. 4(d) verifies that the collaboration among the existing developers is common as a whole, especially with regard to the o1o collaboration. This explanation should be quantified and be properly evaluated advantage of o1o collaboration is that, for an existing developer in our context, the set of candidate collaborators who never cooperate with him/her is larger than the set of his or her prior collaborators. However, compared with the available collaborations among these existing developers, the collaborations discussed above are significantly less frequent. As an OSS ecosystem, the less utilization of existing resource does not seem to benefit its development, in particular from the perspectives of sustainability. It is one of the main reasons why we emphatically focus on the collaborative behavior of existing developers.

For both collaboration types, we also compared the difference between each pair of developers on their joined time when a developer registers in the SF ecosystem. Fig. 4(e) shows that the span of pairwise developer's joined time in terms of o1o collaboration is greater than that in terms of o2o collaboration. More specifically, the majority of o1o collaborations have a more than one year's joined time gap between their developers. The max gap is get up to 8 years in the fifth time interval.

Until now, we have obtained an initial understanding of the distribution of new collaborations among existing developers. In practice, however, one may wonder which two existing developers are more likely to establish the o1o or o2o collaboration in the following time. So it is very necessary to explore that what factors will affect the collaborations among existing developers in an OSS ecosystem.

RQ2: What factors affect the collaborations among existing developers in OSS ecosystem?



(a)Average shortest path length (b) Clustering coefficient (c) Gini coefficient

Fig. 2. The evolution analysis of existing developer networks.

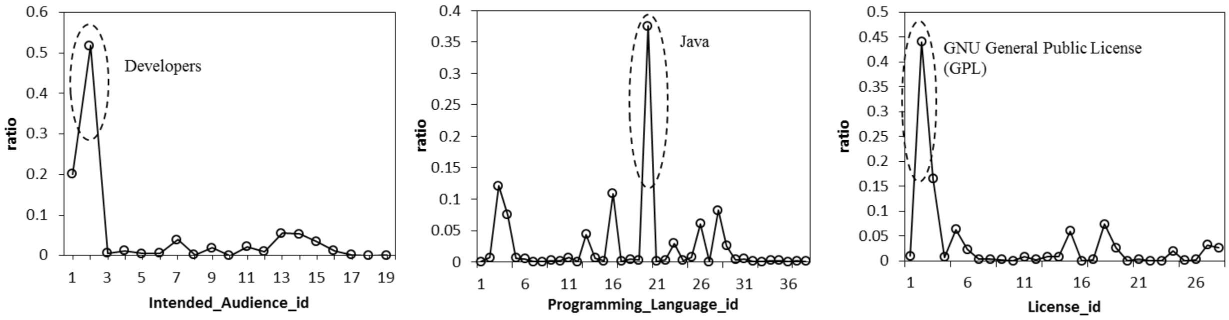
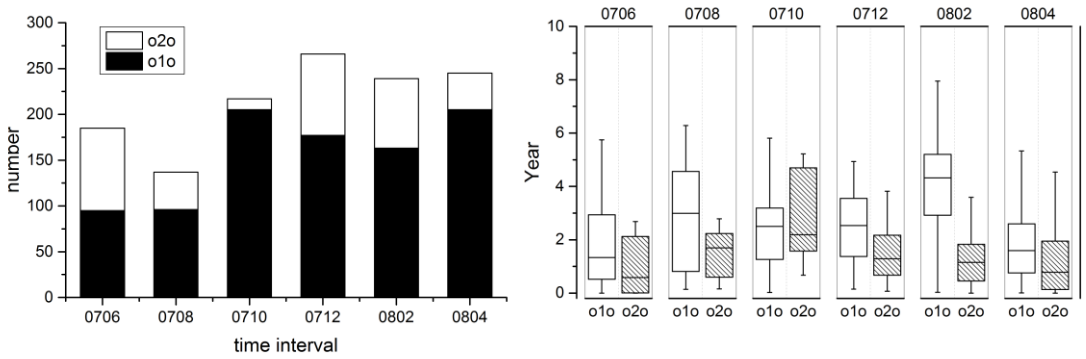
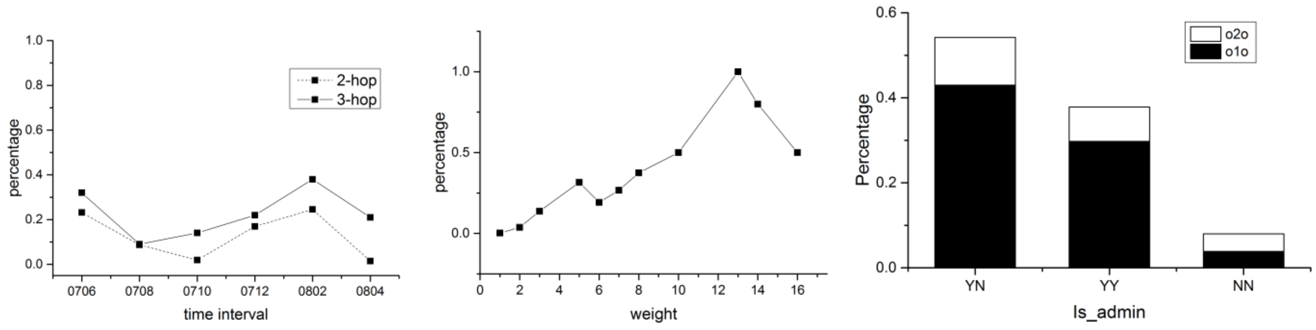


Fig. 3. The distribution of new collaborations among the existing developers according to different attributions. (In each subFig. , the x- axis is the elements' id in database).



(d) New collaborations (e) The difference of joined time

Fig. 4. The distribution of two types of collaborations among the existing developer.



(f) Topological distance (g) Edge weight (h) Administrator experience

Fig. 5. The impact of topological distance and edge weight on the new collaboration, respectively.

Given the EDNs, we first analyzed the impact of network structure on the two types of collaboration patterns.

For the o1o pattern, Fig. 5(f) shows a low percentage of collaboration established based on “friends of friends”. Strictly speaking, in contrast to other general social networks, collaborative relationships established based on the o1o pattern are more likely to be created freely and loosely because of the openness and freedom of the SF community. That is, the strength of social ties (i.e., co-working in a virtual community) is rather weak compared with kin relations and close personal friends. Therefore, short topological distance between existing developers who did not work together before has very limited effect on the first collaboration.

For the o2o pattern, we investigated how edge weight affects the reoccurring collaboration. There is an overall increase on the percentage of new collaborative relationships with an increase of edge weight, especially when the weight is more than 10 (see Fig. 5(g)). It suggests a positive correlation between edge weight and the possibility of repeated collaboration. Thus, good cooperation in the past would be able to strengthen the relationships (e.g., trust) between developers, so that they are willing to collaborate with each other again in the future.

For a developer, the extent of getting access to an OSS project’s resources largely limits his or her contribution to this project. Thus, according to the information provided on whether an existing developer had administrator privilege or not, we further analyzed the distribution of collaborations. Note that, if both existing developers act as an administrator in a project, the attribute Is_admin of their collaboration is labelled as YY, and NN represents neither of them has the relevant experience. Fig. 5(h) shows that more than half of the collaborative relationships are established based on the YN style, followed by YY and NN. However, for the YN and YY styles, compared to the o2o collaboration, the ratios of the o1o collaborations to the total are up to 79.3% and 78.7%, respectively. About 92% of the collaborative relationships contain at least one member who acts as an administrator, indicating that administrator experience may play an important role in determining the future collaboration among these existing developers.

Table 3. The Impact of Correlation Coefficient and Expertise Similarity on the New Collaborations

Time interval	o1o					o2o				
	Sim	***r***	**L-L**	**L-H**	**H-H**	**Sim**	***r***	**L-L**	**L-H**	**H-H**
070	0.899	-0.225	0.221	**0.653**	0.126	0.014	0.707	0.033	0.233	**0.734**
070	0.605	-0.108	0.208	**0.615**	0.177	0.153	0.822	0.098	0.000	**0.902**
071	0.444	-0.691	0.117	**0.646**	0.237	0.327	0.767	0.250	0.000	**0.750**
071	0.756	-0.679	0.147	**0.763**	0.090	0.446	0.876	0.236	0.203	**0.561**
080	0.619	-0.355	0.110	**0.687**	0.202	0.250	0.577	0.066	0.258	**0.676**
080	0.592	-0.725	0.146	**0.771**	0.083	0.184	0.739	0.169	0.240	**0.591**

Sim is the Similarity of Expertise of two Existing Developers, r is the Spearman Correlation Coefficient Based on Developer Collaboration Network. Low($degree(i) \le \mu$) , High($degree(i) > \mu$), $degree(i)$ is the Degree of Developer and is a Specific Value

Finding the relevant expertise is a critical requirement in collaborative software engineering, particularly in geographically distributed developments. We analyzed thus the impacts of developer expertise on the two types of collaborations. Clearly, in Table 3, the similarity of developer expertise for o1o collaboration is remarkable, up to 0.899. However, this factor is not informative for o2o collaboration, indicated by the relatively low values.

On the other hand, the degree-degree correlation coefficients r shows that (see Table 3), for o1o collaboration, there is a negative correlation among developers. That is, a developer with low degree is more likely to select the developers with high degree as candidate collaborators to improve his/her development experience. However, the result of o2o collaboration is opposite; there is a significant trend of preferential collaboration among those developers with high degree, indicated by the strong positive correlation coefficients. The finding accords with the common sense of “win-win cooperation” on social networks, suggesting that experienced developers (viz. nodes with high degree) prefer to work with their peers.

In short, the impact of topological distance between existing developers is limited, but some of developer’s experiences seem to be the main determinations.

RQ3: What are the typical interactive modes among existing developers in our context?

With regard to the skewed distribution of developer degree, we attempted to differentiate the existing developers by dividing them into two groups based on the well-known "eighty-twenty" rule. In each time interval, about twenty percentages of developers have more than 14 collaborators in the current ecosystem.

Therefore, if the degree of an existing developer is not over 14, namely degree (i) ≤14, he is classified into the group Low; otherwise, he belongs to the group High. We then gave a more detailed analysis to explore the typical interactive modes in the two types of collaborations. Table 3 presents the ratio of three possible outcomes for both two types of collaborations. It is very clear that, for o1o collaboration, the L-H mode has a significantly greater value than the other two in each time interval, followed by L-L mode and H-H mode. However, the H-H mode becomes the most frequent one when it comes to the o2o collaboration. Furthermore, there are no o2o collaborations to the L-H mode in two cases.

To better explain the typical interactive mode obtained above (the L-H mode in o1o collaboration and the H-H mode in o2o collaboration), we calculated the number of projects that each type of developer participated in, respectively. Fig. 6 shows that the developers from o2o collaboration labelled as H role tend to participate more projects than those from o1o collaboration labelled as L role and H role. These developers not only have many partners, but also are involved in several projects. In addition, for the L-H mode in o1o collaboration, the developers labelled as H role work on less projects than those labelled as L rule, indicated by the lower median value. In other words, some developers as the core member of certain project never contribute to other projects, but they have a few partners because of the active participation of a large number of developers. On the contrary, there are some developers that they have few partners owing to the small scale of team, although they contribute to several projects. In practice, however, the two types of existing developers mentioned above often develop new collaborations to achieve a complementary effect.

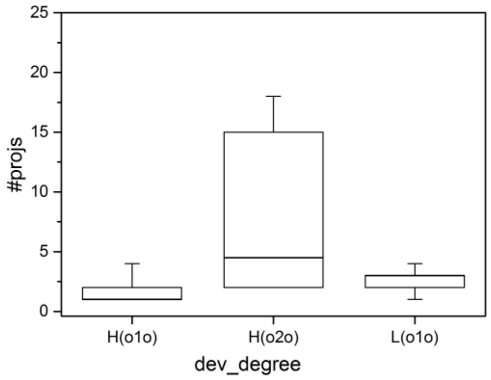


Fig. 6. The number of projects that each developer participated in according to the L-H mode for o1o and H-H mode for o2o.

5. Discussion

RQ1: In this section, other commonly-used measures such as degree distribution are not utilized to present the evolution of EDN in the SF ecosystem. However, similar results have been reported in [34], [35]. As we know, there are many methods for constructing the developer collaboration network. Here, we defined a specific type of coarse-grained networks from the perspective of existing developers' co-working in the same OSS project. Some may argue that the EDN based on co-working are farfetched, but the core developers (a fraction of them) are likely to collaborate with each other due to their frequent interactions, such as by mailing list and CVS. Similar buildings of collaboration networks can be found in some prior studies [34], [36], [37].

The short path length provides more opportunities for existing developers to establish new collaboration, in particular for those who did not work together before. However, the increasing Gini coefficient suggests that the cooperation among the existing developers is preferential. To our knowledge, the skew development is not conducive to the healthy development of an OSS ecosystem to some extent. It is easy to make the development of OSS ecosystem to be extreme, but rather to improve the collective development. Therefore, once we own a comprehensive and detailed understanding of the interactive behavior of existing developers, many effective measures can be taken to support the collaboration among these developers. For example, in order to alleviate the sparse interaction in an OSS ecosystem, one can recommend some potential peers to a target developer according to his specific behavior.

Additionally, the experience as admins not only makes the developer own the special access to the resources, but also attracts more developers to collaborate with those people with this experience. The benefit, to a certain extent, encourages the developers to become more aspiring and skilled through the legitimated peripheral participation (LPP) [38]. In fact, the number of collaborations based on the NN style may greatly increase if the activities of all members are considered. The reason is that the number of core developers just accounts for a fraction of total developers. Besides, the obvious difference between each pair of developers on their joined time further indicates that the selection of suitable partners for a developer is not informal, which deserve further research on their mechanisms.

RQ2: Subsequently, we attempted to investigate the factors that may affect the collaborations among existing developers in OSS ecosystem from the perspectives of topological structure of EDN and developer experiences. The results indicate that “friends of friends” has very limited effect on the o1o collaboration, which is different from the common finding concluded from the general social network. A possible reason for the "friend of friends" relationship does not work in a development context because the limited developers are considered in the EDN and the cooperation between developers in our context is slightly different from the interaction between users in the social networking. It is well-known that the physical distance is no longer a factor affecting the development of OSS. Therefore, given the result obtained in this paper, we may deem that distance (topological and physical) is not a key factor for the o1o collaboration. Although the topological distance between two developers plays a very small role, their expertise measures appear to be very relevant. That is, compared with the advantage of topological structure, a developer is more likely to establish cooperation with another developer who experiences the similar development.

For the o2o collaboration, rich experiences of cooperation in the past and “win-win cooperation” do affect the reoccurring collaboration. From the practical point of view, “win-win cooperation” provides an important insight into the understanding of collaborative development activities and the learning of the major motivational forces in an OSS ecosystem. In software engineering recommendation system, one can recommend appropriate partners for the target developer according this “win-win” requirement. In addition, the collaborations built between two developers with rich cooperative experience are more likely to avoid the activities of low quality, that is, fewer defects will be introduced.

RQ3: There are a wide variety of possible interactive modes that may support existing developer’s collaborative behavior. According to the experiments conducted in this paper, our experimental results also show that developer role is more likely to be the determinant of building potential (first and reoccurring) collaboration between two existing developers by dividing them into different groups. Reasonably, the H-H mode is more frequent for the o2o collaboration because of “win-win cooperation” validated above, and the L-H mode is more suitable for the o1o collaboration due to the learning process and the complementary effect on development experience. Our work is useful to complement this gap by presenting some typical interactive modes among existing developers.

6. Threats to Validity

Threats to construct validity are primarily related to the data sets we used. All of the data sets were collected from the SF ecosystem with the support of Flossmole.org. These data sets have been validated and applied to several prior studies [34], [37], though they only reported the early state of the community. Therefore, we believe that our results are credible and can be reproduced.

Threats to internal validity are mainly related to the measures and various attributes used in our study. For our experiments, we chose several network measures and some attributes as the possible factors that may affect existing developers' collaborative behavior to answer RQ1 and RQ2. In order to answer the RQ3, we also utilize the well-known "eighty- twenty" rule to divide the level of developers. However, we are aware that other findings would be revealed if more factors are involved. Although part of factors have been investigated, the results are still valuable to analyse why existing developers collaborate with each other.

Threats to external validity could be related to the generality of the results in other OSS ecosystems, such as GitHub and Google code. We cannot claim that our results can be generalized absolutely as the ecosystem environment and equipment supported is different. Thus, our findings need to be further validated. This threat can be removed by replicating our study across different topics and environments.

7. Conclusion

This study reports an empirical analysis aiming at understanding how existing developers choose the suitable developers to establish a new (first or reoccurring) collaboration in an OSS ecosystem and at making sense of what factors may affect their collaborative behavior. The experiments have been conducted by taking the Sourceforege.net ecosystem as the object.

In summary, our results show that the impacts of different factors of topological structure and developer expertise on existing developers' collaborative behavior are varied. Short topological distance (known as "friends of friends") does not lead to close collaboration among existing developers, while those edges with great weight and nodes with high degree do (known as "difficult the first time, easy the second" and "win-win cooperation"). In addition, administrator experience and developer expertise seem to play an obvious role in determining the priority of building a potential collaborative relationship between two existing developers. Finally, we also found the typical interactive modes for the two types of collaboration patterns in our context from the perspective of developer degree. Some of our findings could improve the rate of human resource utilization and the maintenance of OSS ecosystem, such as proper peer recommendation for a target developer.

Our future work will focus primarily on two aspects: (1) taking other OSS ecosystem into account, as stated previously, to validate the generality of our findings; (2) analysing more attributes from the data dump of the SF to provide a comprehensive understanding on developers' collaborative behavior.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (Nos. 61273216, 61272111, 61202048 and 61202032), the Natural Science Foundation of Hubei Provincial (No. 2016CFB309).

References

[1] Ye, Y., & Kishida, K. (2003). Toward an understanding of the motivation open source software developers. *Proceedings of the International Conference on Software Engineering*.

[2] Manikas, K., & Hansen, K. M. (2013). Software ecosystems – A systematic literature review. *Journal of Systems & Software*, *86(5)*, 1294-1306.

[3] Madey, G., Freeh, V., & Tynan, R. (2002). The open source software development phenomenon: An analysis based on social network theory.

[4] Lee, S. Y. T., Kim, H. W., & Gupta, S. (2009). Measuring open source software success ☆. *Omega*, *37(2)*, 426-438.

[5] Crowston, K., Annabi, H., & Howison, J. (2003). Defining open source software project success.

[6] Roberts, J. A., Hann, I. H., & Slaughter, S. A. (2006). Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the apache projects.

[7] Singh, V. (2012). Newcomer integration and learning in technical support communities for open source software. *ACM International Conference on Supporting Group Work*.

[8] Ngamkajornwiwat, K., Zhang, D., Koru, A. G., Zhou, L., & Nolker, R. (2008). An exploratory study on the evolution of OSS developer communities. *Proceedings of the, Hawaii International Conference on System Sciences*.

[9] Watson, J. B. (1994). Psychology as the behaviorist views it. *Psychological Review*, *101(2)*, 248--253.

[10] Bird, C., Gourley, A., Devanbu, P., Swaminathan, A., & Hsu, G. (2007). Open borders? immigration in open source projects. *Proceedings of the International Conference on Software Engineering Workshops*.

[11] Murphy, J., Hallinger, P., Peterson, K. D., & Lotto, L. S. (2008). Motivation in software engineering: a systematic literature review. *Information & Software Technology*, *50(9–10)*, 860-878.

[12] Krogh, G. V., Spaeth, S., & Lakhani, K. R. (2003). Community, joining, and specialization in open source software innovation: A case study. *Research Policy*, *32(7)*, 1217–1241.

[13] Fronza, I., Sillitti, A., & Succi, G. (2009). An interpretation of the results of the analysis of pair programming during novices integration in a team. *Proceedings of the International Symposium on Empirical Software Engineering and Measurement* (pp. 225-235).

[14] Herraiz, I., & Robles, G. (2006). The processes of joining in global distributed software projects. *Proceedings of the 2006 International Workshop on GLOBAL Software Development for the Practitioner* (pp. 27-33).

[15] Dagenais, B., Ossher, H., & Bellamy *et al.* (2010). Moving into a new software project landscape.

[16] Ducheneaut, N. (2005). Socialization in an open source software community: A socio-technical analysis. *Computer Supported Cooperative Work*, *14(4)*, 323-368.

[17] Hahn, J., Moon, J. Y., & Zhang, C. (2008). Emergence of new project teams from open source software developer networks: Impact of prior collaboration ties. *Information Systems Research*, *19(3)*, 369-391.

[18] Toral, S. L., Martínez-Torres, M. R., & Barrero, F. (2010). Analysis of virtual communities supporting OSS projects using social network analysis. *Butterworth-Heinemann*.

[19] Crowston, K., Wiggins, A., & Howison, J. (2010). Analyzing leadership dynamics in distributed group communication. *Proceedings of the Hawaii International Conference on System Sciences* (pp. 1-10).

[20] Hossain, L., & Zhu, D. (2009). Social networks and coordination performance of distributed software development teams. *Journal of High Technology Management Research*, *20(1)*, 52-61.

[21] Datta, S., Kaulgud, V., Sharma, V. S., & Kumar, N. (2010). A social network based study of software team dynamics. Proceeding of the, *Proceedings of the India Software Engineering Conference*.

[22] Datta, S., Sindhgatta, R., & Sengupta, B. (2011). Evolution of developer collaboration on the jazz platform: A study of a large scale agile project. *Proceedings of the India Software Engineering Conference* (pp. 21-30)

[23] Huang, S. K., & Liu, K. M. (2005). Mining version histories to verify the learning process of legitimate peripheral participants. *International Workshop on Mining Software Repositories*.

[24] Allaho, M. Y., & Lee, W. C. (2013). Analyzing the social ties and structure of contributors in open source software community. *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining* (pp. 56-60).

[25] Mens, T., Claes, M., & Grosjean, P. (2014). ECOS: Ecological studies of open source software ecosystems. *Software Maintenance, Reengineering and Reverse Engineering*.

[26] Hanssen, G. K. (2011). A longitudinal case study of an emerging software ecosystem: implications for practice and theory. *Journal of Systems & Software*, *85(7)*, 1455-1466.

[27] Gamalielsson, J., & Lundell, B. (2014). Sustainability of open source software communities beyond a fork: How and why has the libreoffice project evolved?. *Journal of Systems & Software*, *89(C)*, 128-145.

[28] Posnett, D., D'Souza, R., Devanbu, P., & Filkov, V. (2013). Dual ecological measures of focus in software development. *Proceedings of the International Conference on Software Engineering* (pp. 452-461).

[29] Singh, P. V. (2010). The small-world effect:the influence of macro-level properties of developer collaboration networks on open-source project success. *ACM Transactions on Software Engineering and Methodology*, *20(2)*, 1-27.

[30] Howison, J., Conklin, M., & Crowston, K. (2006). Flossmole: A collaborative repository for floss research data and analyses. *International Journal of Information Technology & Web Engineering*, *1(3)*, 17-26.

[31] He, P., Li, B., & Huang, Y. (2013). Applying centrality measures to the behavior analysis of developers in open source software community. *Proceedings of the Second International Conference on Cloud and Green Computing* (pp. 418-423).

[32] He, P., Li, B., Yang, X. H., Xiong, W., & Chen, J. (2014). Roster: An approach to potential peer recommendation for developers. *Chinese Journal of Computers*.

[33] Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information Processing & Management*, *24(5)*, 513-523.

[34] Saraf, N., Seary, A., Chandrasekaran, D., & Monge, P. (2012). Evolution of an open source community network. *Advances in Network Analysis and Its Applications*. Springer Berlin Heidelberg.

[35] Weiss, M., Moroiu, G., & Zhao, P. (2006). Evolution of open source communities. *IFIP International Federation for Information Processing*, *203*, 21-32.

[36] Xu, J., Gao, Y., Christley, S., & Madey, G. (2005). A topological analysis of the open souce software development community. *Proceedings of the Hawaii International Conference on System Sciences*.

[37] Gao, Y., & Madey, G. (2007). Network analysis of the sourceforge.net community. *Open Source Development Adoption & Innovation*.

[38] Eberle, J., Stegmann, K., & Fischer, F. (2014). Legitimate peripheral participation in communities of practice: Participation support structures for newcomers in faculty student councils. *Journal of the Learning Sciences*, *23(2)*, 216-244.

Peng He is a lecturer at the School of Computer Science and Information Engineering, Hubei University, Youyi avenue, Wuchang district, Wuhan 430062, Hubei province, China