Network Analysis of a Large Scale Open Source Project

Alma Oručević-Alagić, Martin Höst
Department of Computer Science, Lund University, Sweden
Email: (alma, martin.host)@cs.lth.se

Abstract—**One way to understand the structure of an open source community is by applying network analysis to its source code repositories. In this paper a new method for the analysis of committers' networks is proposed. The method deals with directed and weighted committers' networks. The method is then applied to the Android open source project. The analysis results show how a large, company sponsored, and industry backed open source project, i.e. an open source project with the majority of the community members affiliated with the industry, is structured. In particular, it shows that the involvement of an entire industry eco system within a company sponsored open source project does not imply more equal distribution of the participating community members' influences in terms of committers' networks.**

I. Introduction

Open source software (OSS) has been growing in importance and affecting the way companies develop their products and services [1], plan their business strategy, and compete [2].

There have been many studies conducted on open source projects by analyzing source code change logs and mailing list archives in order to understand the underlying structure and behavior of the community. The studies focused either on some individual open source projects [3], or on an entire portal hosting tens of thousands of open source projects [4].

Android was initially developed as proprietary software by the Android corporation. In 2005, Google Inc. bought Android [5] and open sourced the operating system in 2007 [6]. At the same time, Google also founded the open handset alliance [6]. The open handset alliance is a consortium of over eighty globally leading companies in market segments of mobile operators, handset manufacturers, semiconductors, software, and commercialization companies. The companies contribute to the development of the Android and deliver devices and services built around the Android operating system. Companies like Vodafone, Sprint, T-Mobile, Acer, HTC, Samsung, Sony Mobile, Intel, ST Ericsson, eBay, Accenture, are some of the members of the alliance. Besides the core components open sourced by Google in 2007, Android also includes other open source projects, the majority of which were in existence before the Android project.

The outline of this paper is as follows.

In Section II, the research method is further defined. Section III presents the obtained results, while Section IV discusses and analyses the obtained results in some more detail. Finally, conclusions are drawn in Section V.

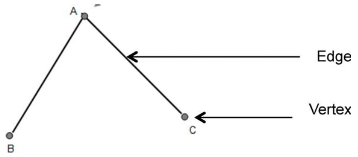

Fig. 1. A Three Actor Network

II. Research Approach

A. Introduction

The study is conducted as a case study [7]. The investigated case is the committers' network structure of the Android OSP. The study is exploratory with the overall objective to understand how the community participants collaborate in development of the software through the Android OSS process. In order to obtain study data, source code change log data was extracted for each file within the Android repository, and loaded into a database to simplify data manipulation process which identified all pairs of authors that modified the same file. For each identified co-authorship pair, weight and direction of the relationship was calculated. Finally, this data was loaded into the Gephi [8] software for network visualization and analysis.

A study by Luis et al. [3] has shown how social network analysis methodologies can be used to study OSS projects in order to characterize the projects' evolution over time as well as the projects' structure. Affiliation networks are a special type of social network where two distinct sets of actors are related, e.g., a committer network relates a set of committers to a set of changed source code modules. Hence, there exists a link between two committers when they have changed a same module. An actor or a network node is referred to as a vertex and the links between the vertices are called edges as shown in Figure 1.

In this study we propose an approach for studying committers' networks. In Luis et al. [3] the proposed methodology establishes links between the committers, where the weight of the link or the edge is calculated as being the number of commits performed by committers to all common modules, i.e. the degree of relationship. The definition of the common module differs between projects, but usually corresponds to the top level directories of a source code repository. In study by Jermakovics et al. [9] weights are based on the number of files users have modified together, i.e. user similarity.

According to Borgatti and Halgin [10] an important factor to consider when studying the strength of the co-affiliation

978-1-4799-5795-8/14 $31.00 © 2014 IEEE
DOI 10.1109/SEAA.2014.50

among an event's participants is the size of the event. The research suggests that one of the ways to normalize the strength of a co-affiliation between event participants is to weight participation relative to the size of event. In the studied context, the size of the event is the total number of changes made to same file. Then, the strength of co-affiliation among participants relative to the size of event can be expressed as the number of file changes performed by each participant relative to the total number of the changes performed on the file by all participants. For example, if two companies, A and B, make changes to the same file, where company A makes only a few changes while company B makes a majority of the changes, then the influence of A over B is much smaller than the influence of B over A. Hangal et. al [11] also examine asymmetric influences of nodes through a friendship example and infers weights on friendship relationships. Hence, we propose a new approach to study committer networks as weighted digraphs as shown in Figure 2. The figure shows weights of the edges for committers associated with companies A, B, and C who have changed the same source file 5, 10, and 15 times, respectively. The edge weight is calculated as the number of the committers' changes on a file relative to the total number of changes for the file, which in this case is 30. Thus, the committer A infers a weighted influence of 1/6, B of 1/3, and C of 1/2 to the files's co-committers.

Toivonen [12] argues the importance of strengths of edge ties when modeling social structure and dynamics of social networks. We argue that inferring the edge weight relative to the size of the event provides a more accurate social network structure compared to the one suggested by Luis et al. [3], which does not take into account the relative size of an event. For example, if only a degree of relationship is considered in the above example for the committers A, B, and C for, e.g., the total number of files they changed together, then the edge weights between the three committers would be the same. This would mean that the strength of co-affiliation between A, B, and C is the same relative to the source file change event, which is clearly not the case. While this is a simple and trivial example, in a context of a large network, with many committers, where, e.g, a subgroup of committers performs a large number of changes, computing edge weights relative to the number of all changes performed on a file is important in order to accurately assess the strength of a relationship. This is more so as the data on committers, corresponding edges, and their weights are building elements of a network structure, based on which other network metrics are derived.

In this study, the weight of the edge between two participants is calculated on a file level. Affiliation networks link actors into a social network by virtue of participants attending a specific event. In the context of committer network analysis we define the event as performing modifications on a specific source code file. Hence, for a set of actors $V = \{v_1, v_2, ..., v_k\}$ and events $U = \{u_1, u_2, ..., u_m\}$ we define a weight W of an edge between an actor v_i and all other actors that participate in the event u_t as:

$$W(v_i, v_j, u_t) = \frac{X(v_i, u_t)}{\sum_{c=1}^{k} X(v_c, u_t)}$$

where $X(v_i, u_t)$ denotes the number of times an actor/committer v_i made changes to the file, i.e., participated in the event u_t.

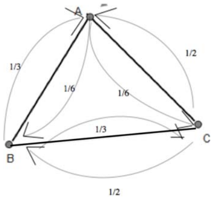

Fig. 2. A Weighted Three Actor Network for Modification of One Source Code File

This means that the weight of the edge $W(v_i, v_j)$ for all events v_i and v_j attended together equals:

$$W(v_i, v_j) = \sum_{t=1}^{m} W(v_i, v_j, u_t)$$

In order to obtain committer data on the source code changes, Android project source code repository was downloaded in November 2012 from the Android project web site [13]. Change log records, with information on authors and change dates for all Android source code files were extracted and loaded into a database. The social network data on network nodes/committers, edges, and associated edge weights and labels was analyzed using the Gephi software for social network analysis [8]. The labels correspond to the main subdirectories under the Android source code tree, as displayed in Table **??**. The Gephi software was used to calculate relevant social network metrics which are discussed in more detail in Section 2.4 as well as to generate a visual representation of the committers' networks. Besides analyzing committer network for the entire repository, we also analyze two additional distinct committer sub-networks. Three committers' subnetworks are constructed:

1) External committers network which includes committers that changed files located under the external top subdirectory.
2) Core committers network which includes committers that changed files located under all top subdirectories excluding the external subdirectory.
3) The entire committers network which includes committers that changed files located under both, the core and the external subdirectories.

B. Research questions

The following research questions were investigated during the research:

1) What are the characteristics of the committers' networks for each set of the Android project source files: the core, the external, and combined core and external?
2) How can a company utilize network analysis to study a development community?

For research question 1, the focus is an assessment of the three distinct network structures, the core components committer network structure, the external committer network structure, and the combined core and the external committer network structure. Metrics on network influence, clustering, centrality, existence of sub-communities, and network density are presented and discussed.

For research question 2, we analyze the results of research question 1 from the perspective of a company planning to develop software through Android or a similar OSP.

C. Investigated software

A program that parses trough all source files located under the Android OSP subdirectories was created and run in order to collect information on all the changes made to all the source files in terms of authors and change dates. The extracted data was loaded into a relational database to simplify manual data validation and to provide a flexible way to create different input formats to the Gephi [8] social network analysis software. All committers were grouped based on a company affiliation. The affiliation is determined based on committer's e-mail domain suffix. In case email data was not provided, committers' individual names are used and no company affiliation is implied. All of the contributions made by the author named "Initial Android Open Source Project Contribution was excluded from the analysis, as these contributions were not developed under the Android OSS community process, but internally by Google before the project was initially open sourced. The Gephi data records are of the form "source, target, edge weight, edge label. If we consider the earlier example depicted in Figure 2, a sample record would look like "A, B, 1/6, the changed source file's top subdirectory.

D. Metrics

The following metrics were measured for the three committer networks:

- Weighted average in-degree *WAID* and weighted average out-degree *WAOD* of a vertex.
- Betweenness centrality *BC*, closeness centrality *CC*, and eigenvector centrality *EVC* of a vertex.
- Average Clustering Coefficient *ACC* of a vertex.
- Modularity *MC* of a network.
- Number of *MCN* of a network.
- Graph density *GD* of a network.

Weighted degree of a vertex denotes degree of relationship of the vertex with its direct neighborhood. It is calculated as the sum of weights of all edges connected to the vertex. Since the analyzed network is weighted digraph, there exist two types of edges; the edges originating from a vertex, or the out degree ($WAOD$), and the edges pointing to a vertex, or the in-degree ($WAID$). In the context of committer network analysis, the out degree can be interpreted as the measure of collaboration strength or influence of the committer on committers in its direct neighborhood.

Betweenness centrality index (BC) is the number of shortest paths that traverse through a vertex and it can be interpreted as a measure of importance of the vertex in a graph. The higher betweenness centrality index of a vertex, the more important the vertex is.

Closeness centrality CC indicates how close on average a vertex is to all other vertices. A high value of the distance centrality index identifies vertices that are well related.

Eigenvector centrality EVC metric measures the influence of a vertex on a network by assigning scores to all vertices in the network.

textitAverage clustering coefficient ACC of a vertex shows the tendency of the network to form cliques or isolated groups.

Modularity of a network MC identifies the sub-communities within the network with densely connected vertices. The value of modularity is calculated as a difference in fraction of edges that fall into the sub-communities and a fraction of edges that could be found in the sub-communities if the edges were distributed at random per Blondel et. al [14]. In the context of the committer network study, the modularity class is used to identify committer sub-networks with higher degree of collaboration.

Graph density index GD measures how close the network is to being complete, i.e., that there exist edges between all the vertexes in the network. A value of 1 for the graph density index indicates a fully complete or connected network.

E. Analysis procedure

Analysis with respect to research question 1 was conducted by calculating the $WAID$, $WAOD$, BC, CC, EVC, ACC, MC, GD on the core, external, and combined core and external Android OSP source code tree. For research question 2, the presented network structure data in question 1 is analyzed from a business/company perspective.

F. Validity

In this section the validity of the research is analyzed with respect to the types of validity threats presented, for example, in [7].

Construct validity: The construct validity is related to the relationship between the concepts and theories behind the experiment and what is measured and affected. The subset of metrics from the network theory used in this research has been accepted and validated in other studies within the filed of OSP repositories and mailing archive studies. This means that the risk of using metrics that do not represent the concept of social network structure is lowered.

Conclusion validity: The conclusion validity is concerned with the possibility to draw correct conclusions regarding the relationship between treatments and the outcome of an experiment. The interpretation of the metrics is grounded in the widely accepted network theory and the field of social network analysis.

Internal validity: The internal validity is concerned with factors that may affect the dependent variables without the researcher's knowledge. The data extracted from the repositories was examined and validated manually through sampling. The approach used in constructing committers network is grounded on network theory concepts applied in other disciplines.

External validity: The external validity is related to the ability to generalize the results of the experiments. The studied software is a relevant example of a successful industry led OSP as the project includes leading global companies from the mobile eco system.

TABLE I. SUMMARY OF THE COMMITTERS' NETWORKS MEASURES

Metric	Core	External	Core and External
ACC	0.782	0.791	0.799
MC	0.0009	0.356	0.43
MCN	4	6	7
GD	0.058	0.104	0.055

III. Results

A. *Research question 1: What are the characteristics of the Android committers' networks?*

This section provides an assessment of the three distinct network structures, the core components committers' network structure, the external committers' network structure, and the combined core committer's and the external network structure.

The core committers' network has a total of 250 vertices and 3606 edges, which in this case means that committers have 250 distinct affiliations and there are 1803 distinct committer co-authorship pairs. Since the network is modeled as a weighted digraph, the edges are bi-directional. The external committers' network has 329 vertices and 11196 edges, while the combined core and external committers' network has 513 vertices and 14484 edges.

Table I shows ACC, MC, MCN, and GD for the three studied committer network structures. The average clustering coefficients for the three networks show that networks have a high tendency to form cliques.

The identified number of closely related sub-communities MCN for the core committer network is 4. However, the MC value of 0.0009 indicates that a probability of such sub-communities occurring at random is very high. Hence, the identified potential sub-communities for the core committer network should be disregarded since likelihood of their existence is not significant. The number of sub-communities identified within the external committer network is 6 with the MC value of 0.356 indicating that the likelihood of the existence of the 6 subnetworks is significant. The number of identified sub communities for the combined, external and core committers' networks is 7, with the MC value of 0.43 indicating that the likelihood of existence of the sub-communities is significant.

The graph density metric GD for the core, external, and combined core and external committer networks is 0.058, 0.104, and 0.055, respectively. The value of 1 for GD indicates that all the components within the network are highly connected. Hence, all three types of the committers' network showing low graph density values indicate that the committers' networks are weakly connected. The high clustering coefficient shows that even though many edges between the committers are absent, committers in a direct neighborhood of a committer are well linked.

The $WAOD$ and $WAID$ metrics indicate that for the entire Android source code base Google has the highest strength of co-affiliation with members in its direct neighborhood. However, in the external committers' network, Apple has the highest strength of co-affiliation with the members in its direct neighborhood. Hence, the metrics in summery show:

- **Android core committers network:** The high average clustering coefficient and low graph density indicate that committers in a direct neighborhood of a committer are well linked. The majority, i.e., 40% and 50% of shortest paths between two committers within the core committer network pass trough committers associated with Google.com and Android.com email addresses, respectively.
- **Android external committers network:** The number of sub-communities identified within the external committer network is 6 with the *MC* value of 0.356 indicating that likelihood of the existence of the 6 subnetworks is significant. The *EVC* values for external committers' network is balanced among the top 30 committers. The *BC* value is highest for the committers associated with the google.com address, followed by the commuters associated with the gmail.com, debian.org, nondot.org, and applec.com. Committers associated with the apple.com have the highest influence or collaboration strength.
- **Android core and external network:** The number of identified sub communities for the combined, external and internal committers' networks is 7, with the *MC* value of 0.43 indicating that the likelihood of the existence of the sub-communities is significant. Committers affiliated with the Google email address have the highest value of *WAOD*. Apple has some 30% lower value of *WAOD*, followed by committers associated with gmail.com, nondot.org, and android.com. Values for *CC* and *EVC* are balanced between the top 20 committers, while the *BC* values indicate that some 40% of the shortest paths traverse through committers associated with a google.com, followed by gmail.com with some 20% of the shortest paths, and intel.com with some 2% of the shortest paths.

Based on the results of the analysis, the most influential committers with respect to the strength of collaboration are committers affiliated with google.com and apple.com. Committers affiliated with google.com are also the most influential for the core and combined committers' networks, while the committers affiliated with apple.com are the most influential in the external committers' network. The three committers' networks have low GD values, indicating that the committers are not well connected. The external committers network, composed of over 150 different OSPs has a twice as high value for the GD metric as for the Android core components network. While the network centrality metrics are balanced between the top 20 committers, they decrease sharply for the other project committers. Among the 20 committers with highest $WAOD$ and $WAID$ measures only three are found in both the core committers' network and the external committers' network. The three are google.com, android.com, and gmail.com affiliated committers. Among the 20 committers with the highest centrality measures, only two can be observed in both the core and the external committers' networks. The two are gmail.com and google.com affiliated committers.

B. *Research question 2: How can a company utilize network analysis to study a development community?*

Based on the results presented for research question 1, the Android OSP exhibits characteristics of a highly centralized OSP, where committers with affiliations to google.com, gmail.com, android.com, and apple.com have the highest level of influence. The Android committers' networks have low graph densities, i.e. low connectedness of committers, indicating low cross-collaboration among committers.

From a perspective of a company that is planning to participate or participates in Android or a similar OSP this means that it should take into consideration that OSS product development tends to be highly influenced by one company. This might indicate that the company planning to incorporate the Android into its product will need to work closely with Google to ensure that the changes it needs to see implemented in the source code base are included in a future OSS product release. Google has built different sales models around the Android, primarily the GooglePlay store, the application market for Android devices, AdMob platform, and Web search. Hence, it is in the Google's interest to have the Android used and distributed on as many mobile devices as possible. However, the company should be aware that sales and marketing models change, and different alliances form. In order to influence and lead a large open source project, a company controlling the project development usually has a large development effort dedicated to the project. In case a company is no longer able to support the development it is possible that some other company, or a group of companies takes the lead. Hence, it is possible that the new leader takes the open source project in a direction that might be unfavorable to other project participants.

IV. Discussion

Based on the social network structure analysis results for Android committers's networks, it is evident that Google has the highest degree of influence and centrality. This shows how a large company with significant resources can create a large scale software products using other OSS components. In a company sponsored open source project the company invests a large development effort into the OSS product and there exists a possibility that the company might not be able to maintain the high level of development commitment. This possibility would also mean uncertainty for the future of the OSS product development, and, if realized, it can bring shifts in committers' influence on the project. This can create uncertainty on the future of the OSS product development, an important factor that should be considered by companies planning to join similar company sponsored projects. A company might decide to also closed source and license the open source product. Such situation can then create a vendor lock-in effect, which contradicts a generally accepted notion of an OSS software product being free from vendor lock-in, i.e., users of OSS being less dependent on a software producer.

For any company planning to join an OSP or base its product around an OSP, there is a value in understanding the underlying social structure, especially in terms of the most influential community members, a level of cross-collaboration among the committers, and the existence of subgroups.

V. Conclusion

The conducted analysis have shown that Google has the major influence on the Android OSP. While it is favorable to use an OSS product as a commodity software, and thus decrease development costs by focusing available resources on developing differentiating parts of a product, at the same time this can raise many uncertainties. The future of OSS product whose development is highly sponsored and influenced by one company can come under the influence of market conditions the company finds itself in. This seems to go against the nature of OSS, which among other characteristics includes protection from vendor lock-in, i.e., high dependance of companies using Android on the Google.

More research is needed to understand and properly categorize different OSPs in a way that would help the industry better understand own strategic position in a context of using an OSP to build business model. The research approach proposed in this study can be used as one way of studying a committer's network structure of a software development community.

Acknowledgment

This work was funded by the Industrial Excellence Center EASE - Embedded Applications Software Engineering, (http://ease.cs.lth.se)

References

[1] M. Höst and A. Oručević-Alagić, "A systematic review of research on open source software in commercial software product development," *Information & Software Technology*, vol. 53, no. 6, pp. 616–624, 2011.

[2] E. S. Raymond, *The Cathedral and the Baazar*. O'Reilly Media, Inc., 2001.

[3] L. López-Fernández, G. Robles, J. M. González-Barahona, and I. Herraiz, "Applying social network analysis techniques to community-driven libre software projects," *International Journal of Information Technology and Web Engineering*, vol. 1, no. 3, pp. 27–48, 2006.

[4] J. Howison, K. Inoue, and K. Crowston, "Social dynamics of free and open source team communications," in *Open Source Systems, IFIP Working Group 2.13 Foundation on Open Source Software*, 2006, pp. 319–330.

[5] AndroidCorp, "Google buys android for its mobile arsenal," http://www.businessweek.com/stories/2005-08-16/google-buys-android-for-its-mobile-arsenal, 2005.

[6] AndroidOS, "Breaking: Google announces android and open handset alliance," http://techcrunch.com/2007/11/05/breaking-google-announces-android-and-open-handset-alliance/, 2007.

[7] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, vol. 14, pp. 131–164, 2008.

[8] Gephi, "Open source software for exploring and manipulating networks," https://gephi.org, 2013.

[9] A. Jermakovics, A. Sillitti, and G. Succi, "Exploring collaboration networks in open-source projects," in *OSS*, 2013, pp. 97–108.

[10] S. P. Borgatti and D. S. Halgin, "On network theory," *Organization Science*, vol. 22, no. 5, pp. 1168–1181, 2011.

[11] S. Hangal, D. MacLean, M. S. Lam, and J. Heer, "All friends are not equal: Using weights in social graphs to improve search," *SNAKDD-2010: 4th SIGKDD Workshop on Social Network Mining and Analysis*, ACM, 2010.

[12] R. Toivonen, J. M. Kumpula, J. Sarmaki, J. P. Onella, J. Kertesz, and K. Kaski, "The role of edge weights in social networks: modeling structure and dynamics," *Proc. International Society for Optics and Photonics, SPIE*, vol. 6601, 2007.

[13] Google Inc, "Android open source software project," http://www.android.com/, April 2013.

[14] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large network," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 10, p. P100, 2008.