Identifying the Most Valuable Developers using Artifact Traceability Graphs

H. Alperen Cetin
Bilkent University
Ankara, Turkey
alperen.cetin@bilkent.edu.tr

ABSTRACT
Finding the most valuable and indispensable developers is a crucial task in software development. We categorize these valuable developers into two categories: connector and maven. A typical connector represents a developer who connects different groups of developers in a large-scale project. Mavens represent the developers who are the sole experts in specific modules of the project.

To identify the connectors and mavens, we propose an approach using graph centrality metrics and connections of traceability graphs. We conducted a preliminary study on this approach by using two open source projects: QT 3D Studio and Android. Initial results show that the approach leads to identify the essential developers.

KEYWORDS
software and its engineering → Programming teams.

1 RELATED WORK

Some previous studies examined developer types from different perspectives. Kosti et al. [15] investigated archetypal personalities of software engineers. They chose extraversion and conscientiousness as their main criteria and focused on the binary combinations of them. Cheng and Guo [7] made an activity-based analysis of Open Source Software (OSS) contributors, then adopted a data-driven approach to find out dynamics and roles of the contributors. Also, Crowston et al. [9] examined core and periphery of OSS team communications. Joblin et al. [13] made an empirical study on core and peripheral developers by using count and network metrics. They established evidence that the network-based operationalizations of developer roles outperform count-based operationalizations.

Besides developer types, researchers worked on communication networks of developers. Wu and Goh [19] studied the long-term effects of communication patterns on success, and performed experiments about how graph centrality, graph density and leadership centrality affect the success of OSS projects. Moreover, Allaho and Lee [4] conducted a social networks (SN) analysis on OSS projects, and found that OSS SNs follow a power-law distribution which means a small number of developers dominates the projects. The researchers, also, tried to find how many developers have to quit the project before a it is incapacitated (i.e. truck factor) [5] [8] [18].

3 APPROACH

3.1 Constructing Artifact Traceability Graph
Artifact traceability graphs include software artifacts and the connections among them. We denote nodes for developers, change sets (e.g. commits in Git), source files and undirected edges for commits, reviews, includes relations. Developers can commit or review change sets, and change sets can include a set of source files. Figure 1 shows a sample traceability graph.
We mined QT 3D Studio Git repository [1] between Oct 6, 2017 and Jul 6, 2018. The dataset contains 33 developers, 941 commits and 26872 source files from between Oct 24, 2008 and Jan 26, 2012. We constructed the graphs as described in Section 3 by using NetworkX 2.3 [2] for both case studies. Table 1 shows the scores.

For QT 3D Studio, Dev1 and Dev2 have large betweenness centrality scores. Because these scores are calculated on the developer relation graph which has 86% graph density, they show that many shortest paths pass through Dev1 and Dev2. Therefore, they connect people who do not have any strong direct relationship, and they are classical connectors. When it comes to maveness, Dev4 has the largest score. He is the sole committer of 36% of the files in the project. Because of this we argue for that he is an indispensable member of the team and a typical maven.

Showing that the same metrics work for a project with a large number of artifacts is important to validate our approach. In Android case study, the density of the developer relation graph is 20% that is to say most developers have no direct relation in the graph and the betweenness scores are high for many developers. There are 13 developers having betweenness scores between 0.11 and 0.55, but we can say that the first 4 developers Dev5, Dev6, Dev7 and Dev8 are the connectors because of their higher betweenness centrality values. Also, %64 of source files in the project are committed by Dev9, Dev10 and Dev11; therefore, they are mavens.

There is no ground truth or findings about the same categorization on the same datasets to compare our results, however we can say that our findings correspond to the conclusions of Allaho and Lee [4]. They claim that the OSS SNs follow a power-law distribution, and our results show that a small group of developers pervade and dominate the project. Also, Agrawal et al. [3] assert that most projects, particularly medium to large projects, depend on hero developers which are mavens and connectors in our case.

5 FUTURE WORK

Although we conducted two case studies with thousands of commits and hundreds of developers, our method and metrics cannot be generalized. As future work, we recommend to (1) make further case studies for both OSS and proprietary projects by using similar categorization described in this paper and datasets with more artifact data such as issue and test history; (2) conduct a survey among developers to find the golden set of mavens and connectors to validate this approach; (3) cluster developers using traceability graph and find categories by interpreting the clusters; and (4) use graph convolutional networks to classify developer nodes in a semi-supervised manner [14].
REFERENCES