Categorizing Code Review Result with Social Networks Analysis: A Case Study on Three OSS Projects

Xin Yang[1,a)] Norihiro Yoshida[2,b)] Kenji Fujiwara[1,c)] Yong Jin[1,d)] Hajimu Iida[1,e)]

Abstract: Due to the distributed collaborations and communication nature of Open Source Software (OSS), OSS peer review differs from traditional industry software peer review. In this study, we investigate the relationship between network position of OSS peer review contributors and the outcome of their review results by using social network analysis (SNA). The results provides hints on how network position of contributors affect peer review quality, which can help to understand how OSS developers work and communicate together.

Keywords: Open Source Software, Software Peer Review, Social Network Analysis

1. Introduction

Software peer review refers to the code inspections by developers, rather than the author himself. It can be regarded as one of the most important activities for software development [1], [2]. software project developers use peer review for two main benefit: reducing the defects and saving the cost. Unlike industry projects, most of the Open Source Software (OSS) projects have geographically distributed development environment and there is very few chance for developers to have face-to-face communication. Unless changing the peer review approach, the industry meeting-based peer review seems hard to be applied into OSS projects. As a result, OSS projects hope experienced developers could share their knowledge with new members by review their code. Furthermore, they hope to have a good development community to sharing those experience and knowledge. We investigate some related studies has been done in OSS peer review [8], [9], to the best of our knowledge, this is the first research constructing social networks from mining a peer review repository and also performing social network analysis to study the OSS peer review process.

In this study, we investigate the importance of OSS contributor in peer review process from social perspective. We use social network analysis (SNA) to perform our study bug tracking system. We applied this approach for peer review system to generate peer review social networks, which named: PeRSoN (Peer Review Social Network). Then, we analyzed the Android Open Source Project (AOSP), Qt and OpenStack as case study. Our previous results gave us hints about the relationships among OSS peer review contributor roles, their activities, and the network structure. In this study, we want to investigate the relationship between network position of contributors and the outcome of their review results.

Our main research questions can be summarized as follows:

RQ1 As authors of code patchset, is there any relationship between their code quality and their network position?

RQ2 As reviewer of code patchset, is there any relationship between their review quality and their network position?

In experiment phase, first we extracted data set from OSS peer review history then generated the network - PeRSoN. Then we applied SNA and statistical analysis to address our research questions. We started from the most standard social network measures as centrality metrics, which represent the importances to measure the network position of each contributor. Then we performed experiment for contributor activities and the outcome of their review related activities, For example, we want to investigate that an author in an important network position will contribute high quality code patchset or not, and an important reviewer always give a high quality review or not.

[1] Nara Institute of Science and Technology
[2] Nagoya University
[a)] kin-y@is.naist.jp
[b)] yoshida@ertl.jp
[c)] kenji-f@is.naist.jp
[d)] jin.yong.jq0@is.naist.jp
[e)] iida@itc.naist.jp

2. Related work and background

Some study on OSS peer review has been done in recent years. Rigby et al. examined Apache Server Project and created some metrics similar to traditional inspection in order to find an efficient and effective OSS review technique [8]. They also have studied the broadcast nature of OSS peer review, which is different with traditional peer review method [9]. Some study also suggested use review bot to reduce human effort and improve review quality [3]. In our study, we analyze the review community and relationship between contributors by the analysis of their social network. Social network is a network structure which the vertices represent people or groups of people, and the edges represent social interaction between them such as conversation or notification [7]. We seek to investigate the potential relationship between contributors' activities and their importance in the social network. By applying some standard centrality measures, we analyzed our PeRSoN from the social aspect. Freeman defined three centrality measures: Degree, Closeness and Betweenness, which represent the importance of vertex from different perspective [4].

3. Approach

The traditional research on software engineering is always focus on source code metrics and development history, but we applied a novel approach from social perspective to study peer review which is an important phase in software development. Our dataset come from AOSP, Qt and OpenStack. All these projects use Git to manage the source code and Gerrit to manage peer review. The main approach consisted of 3 main steps:

1) Preparation before experiment. Before the experiment, we extracted the raw review dataset from the Gerrit servers of each project. The detail of mining review repository approach can be found from the study of Hamasaki et al. [5]. Our raw data set is available to download [*1]. Because we focus on social aspect of OSS peer review, the main dataset that we extracted is the contributors information and their activities history. In order to investigate the common of contributors, we grouped the contributors into several role groups by their different review activities.

2) PeRSoN Generation. After we have the contributors information and their review history, we started to generate the peer review social network. We applied an approach which is used for generating bug report network [6]. Unlike their approach, our networks were unweighted and we reduced networks by vertices degree. Based on the broadcasting nature of OSS peer review, we assumed that the contributors who appeared in the same review report must have communication with each other. Thus, we connected all the contributors who participating in the same report. Because we are interested in those contributors who performed more activities in peer review, we reduced the network by removing the vertices which have very few degree.

3) Analysis. We performed social network analysis (centrality measure) for each contributors and analyzed the relationship between contributors centrality measures and their review outcome. We applied several standard centrality measures to measure the importance of contributors from three different perspective. We separated contributors into different groups as authors and reviewers because they have different definitions on the high quality contributions.

[*1] http://sdlab.naist.jp/reviewmining/

参考文献

[1] Ackerman, A. F., Fowler, P. J. and Ebenau, R. G.: Software inspections and the industrial production of software, *in Proceedings of a symposium on Software validation: inspection-testing-verification-alternatives*, pp. 13–40 (1984).
[2] Ackerman, A., Buchwald, L. and Lewski, F.: Software inspections: an effective verification process, *IEEE Software*, Vol. 6, No. 3, pp. 31–36 (1989).
[3] Balachandran, V.: Reducing human effort and improving quality in peer code reviews using automatic static analysis and reviewer recommendation, *in Proceedings of the 2013 International Conference on Software Engineering*, ICSE '13, pp. 931–940 (2013).
[4] Freeman, L. C.: Centrality in social networks conceptual clarification, *Social Networks*, Vol. 1, No. 3, pp. 215–239 (1978-1979).
[5] Hamasaki, K., Kula, R. G., Yoshida, N., Cruz, A., Fujiwara, K. and Iida, H.: Who does what during a code review? datasets of OSS peer review repositories, *in Proceedings of the Tenth International Workshop on Mining Software Repositories*, IEEE Press, pp. 49–52 (2013).
[6] Hong, Q., Kim, S., Cheung, S. and Bird, C.: Understanding a developer social network and its evolution, *in Proceedings of the 2011 27th IEEE International Conference on Software Maintenance*, IEEE Computer Society, pp. 323–332 (2011).
[7] Newman, M.: *Networks: An Introduction*, Oxford University Press, Inc. (2010).
[8] Rigby, P. C., German, D. M. and Storey, M.-A.: Open source software peer review practices: a case study of the apache server, *in Proceedings of the 30th international conference on Software engineering*, pp. 541–550 (2008).
[9] Rigby, P. C. and Storey, M.-A.: Understanding broadcast based peer review on open source software projects, *in Proceedings of the 33rd International Conference on Software Engineering*, pp. 541–550 (2011).