Socio-Technical Aspects of package dependencies in software ecosystems

Anonymous Author(s)

ABSTRACT

I think here we dont need an abstract cause it is a two page abstract. but I dont know how should section it.

1 INTRODUCTION

With the advent of open source components since 1990's there is an ever increasing number of open source libraries which is used either as a part of open source or closed source softwares and developers mostly rely on thirdparty components instead of writing a new one. When a developer depends on another components there is an expectation of change and improvement from the user side. This means software contributors should have social interactions with each other to communicate their issues and solve the problem which is very common in open source communities. Millions of comments and issues on Github is a clear sign of this claim.

On the other side these social interaction of contributors may contain very many beneficial information but little is known about them. We use social interaction to human action related to use of third party software packages in application development process. Studying the way and pattern of communication and behaviour of contributors of software packages can help us to create a profile for each contributor which includes a lot of information about them and based on that we can make important decisions in development process. In fact a developer can decide to depend on a package according to behaviour of the contributors or stop using it after a while because of that. We can consider commenting activity of contributor of some package as a source of information to estimate the consistency of the package we are going to start using it.

Several studies have been proposed about social aspects of software ecosystems for example how developer teams interact and evolve [4], how newcomers progress in a software project [11, 12], how the core team grows over time [6], how developer teams get renewed [1], and how socio-technical patterns affect software success or failure [7]. Some study used Socio-technical metrics to define developer profiles and use them for specific recommendations [3], To reveal how bad social interaction practices can decrease software quality and team productivity [10]. The notion of social debt has been used to evaluate the negative impact of community smells on industrial software [8]. Team diversity factors have been shown to positively impact the team's productivity [9]. How social interaction occurs through issues and feature requests [5]. In this research we proposed an imparical study on relationship between the dependencies between software packages and the contributors of those packages. It means we want to explore socio-technical aspect of software ecosystems. We considered 3 exploratory research question to find if there is some tendency or correlation between package dependency network and the collaborator communication network:

- RQ1 Are contributors of comments more likely to contribute to packages they depend on than other packages?
- RQ2 Is there some relation between the commenting activity and the decision to start depending on/contributing to a package?
- RQ3 is there some relation between the commenting activity and the decision to stop depending on/ contributing to a package?

2 METHODOLOGY

We first selected Cargo package manager as an ecosystem in which we can access the interaction of package maintainers through github or other repositories and then we downloaded their information including their comment activities, commits, pull requests, and issues. It was a time consuming step because there were some restrictions from the Github website and also there were more than 1 milion interaction for almost 10000 packages inside this repostiory. We also gathered meta data about each package release such as its name, its release data, its version number, its maintainer, metadata about package dependencies: which version of which package depends on another package and with which version constraint from libraries.io dataset.

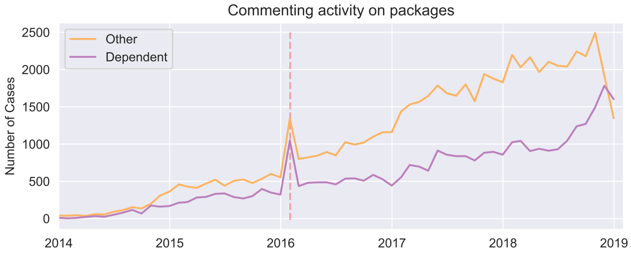
Then we started to analyse and compare the package dependency network and the communication network of contributors of packages. Our aim was to find an answer for the 3 research question we mentioned in previous section. Intutively according to Conway's law [2] we guess if there is some technical dependency between software packages, it will be more likely that there is also a collaboration between package contributors (through commits, pull requests, and their associated comments).

Based on this hypothesis we started to emparically analyse data regarding their package dependency and contributors activity and our findings was interesting and unique which we will discuss in findings section with more detail.

In next section we have some new terminologies to talk about how packages and contributors linked to each other in a social or technical network. We use the term 'Dependency' Where a software component uses another package inside it and 'Reverse dependency' where a package is used as a required package in another software component.

3 ANALYSIS AND RESULTS

RQ1 - According to our findings there are less comment on packages with technical dependency than others but comments on packages which are dependent to each other are increasing as we move toward 2019. It seems that this tendency is going to grow more in recent years. Our investigation shows, the peak in Febraury 2016 caused by a major problem with 2 popular component which was led to a burst of comment by both users and contributors of those packages.



REFERENCES

[1] Eleni Constantinou and Tom Mens. 2017. Socio-technical evolution of the Ruby ecosystem in GitHub. In *IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*. 34–44. https://doi.org/10.1109/SANER.2017.7884607

[2] M. Conway. 1968. How do Committees Invent? *Datamation Journal* (April 1968), 28–31.

[3] Ghadeer A. Kintab, Chanchal K. Roy, and Gordon I. McCalla. 2014. Recommending Software Experts Using Code Similarity and Social Heuristics. In *Proceedings of 24th Annual International Conference on Computer Science and Software Engineering (CASCON '14)*. IBM Corp., Riverton, NJ, USA, 4–18. http://dl.acm.org/citation.cfm?id=2735522.2735526

[4] Luis Lopez-Fernandez, Gregorio Robles, Jesus Gonzalez-Barahona, and Israel Herraiz. 2009. Applying Social Network Analysis Techniques to Community-driven Libre Software Projects. In *Integrated Approaches in Information Technology and Web Engineering: Advancing Organizational Knowledge Sharing*. IGI Global, Chapter 3, 28–50. https://doi.org/10.4018/978-1-60566-418-7.ch003

[5] M. Palyart, G. C. Murphy, and V. Masrani. 2018. A Study of Social Interactions in Open Source Component Use. *IEEE Transactions on Software Engineering* 44, 12 (Dec 2018), 1132–1145. https://doi.org/10.1109/TSE.2017.2756043

[6] Gregorio Robles, Jesus M. Gonzalez-Barahona, and Israel Herraiz. 2009. Evolution of the core team of developers in libre software projects. In *Int'l Conf. Mining Software Repositories*. IEEE Computer Society, 167–170.

[7] Didi Surian, Yuan Tian, David Lo, Hong Cheng, and Ee-Peng Lim. 2013. Predicting Project Outcome Leveraging Socio-Technical Network Patterns. In *European Conf. Software Maintenance and Reengineering*.

[8] Damian A. Tamburri, Philippe Kruchten, Patricia Lago, and Hans van Vliet. 2015. Social debt in software engineering: insights from industry. *J. Internet Services and Applications* 6, 1 (2015), 1–17. https://doi.org/10.1186/s13174-015-0024-6

[9] Bogdan Vasilescu, Daryl Posnett, Baishakhi Ray, Mark G. J. van den Brand, Alexander Serebrenik, Premkumar T. Devanbu, and Vladimir Filkov. 2015. Gender and Tenure Diversity in GitHub Teams. In *Int'l Conf. Human Factors in Computing Systems (CHI)*. ACM, 3789–3798. https://doi.org/10.1145/2702123.2702549

[10] M. S. Zanetti, I. Scholtes, C. J. Tessone, and F. Schweitzer. 2013. The rise and fall of a central contributor: Dynamics of social organization and performance in the Gentoo community. In *Int'l Workshop on Cooperative and Human Aspects of Software Engineering*. 49–56. https://doi.org/10.1109/CHASE.2013.6614731

[11] Minghui Zhou and Audris Mockus. 2011. Does the initial environment impact the future of developers?. In *Int'l Conf. Software Engineering*. ACM, 271–280. https://doi.org/10.1145/1985793.1985831

[12] Minghui Zhou and Audris Mockus. 2012. What make long term contributors: willingness and opportunity in OSS community. In *Int'l Conf. Software Engineering*. IEEE Press, 518–528.