Analysing Socio-technical Congruence in the Package Dependency Network of Cargo

Mehdi Golzadeh
Software Engineering Lab, UMONS
Mons, Belgium
mehdi.golzadeh@umons.ac.be

ABSTRACT

Software package distributions form large dependency networks maintained by large communities of contributors. My PhD research will consist of analysing the evolution of the socio-technical congruence of these package dependency networks, and studying its impact on the health of the ecosystem and its community. I have started a longitudinal empirical study of Cargo's dependency network and the social (commenting) and technical (development) activities in Cargo's package repositories on GitHub, and present some preliminary findings.

CCS CONCEPTS

• **Human-centered computing → Empirical studies in collaborative and social computing**.

KEYWORDS

Socio-Technical congruence, Software ecosystem, Software repository mining, Software development, Package dependency network

ACM Reference Format:
Mehdi Golzadeh. 2019. Analysing Socio-technical Congruence in the Package Dependency Network of Cargo. In *Proceedings of the 27th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE '19), August 26–30, 2019, Tallinn, Estonia.* ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3338906.3342497

1 INTRODUCTION

Today's software development is increasingly relying on software package libraries distributed through open source software (OSS) package managers (such as Cargo, npm, Maven and CRAN). Rather than writing software from scratch, developers often choose to depend on existing software packages. At the same time, collaborative online development platforms like GitHub make software development an inherently social phenomenon [8, 21].

The collection of packages distributed by a software package manager forms a *socio-technical* dependency network. Packages depend on other packages that are required for installing and deploying them. Software developers are *technically* contributing to these packages (e.g., by making commits, pull requests to the package's git repository). Software developers are also *socially* active (e.g., by commenting on the commit and pull request activities). The phenomenon of socio-technical congruence (a.k.a. Conway's law) [7, 13] assumes that the package dependency network structure and the communication structure of its community of contributors are tightly interwoven. Very little research focuses on such socio-technical congruence at the ecosystem level [22], or studies show the congruence evolves over time [5].

My PhD research aims to empirically study, within evolving OSS ecosystems, the socio-technical congruence between package dependencies, the interaction patterns of package contributors and how this affects the health of the ecosystem. To do so I will explore 3 research questions: **RQ**$_1$ *How does the dependency network structure influence social activity?* **RQ**$_2$ *How does the social activity of package contributors increase their likelihood to start/stop depending on this package?* **RQ**$_3$ *How does the social activity of package contributors increase their likelihood to start/stop becoming technically active?*

In the first phase, I will focus on packages distributed through the Cargo package manager and developed on GitHub by studying their *technical* development activity (e.g., GitHub commits and pull requests) and *social* communication activity (e.g., commit comments and pull requests comments). Other activity types and packaging ecosystems will be explored in a later phase, on the basis of the results obtained in the first phase. These questions will focus on the expected benefits that socio-technical congruence will have on the health of the ecosystem and its community, such as increased productivity, responsiveness, contributor intake and retention.

2 BACKGROUND

Software ecosystems are large collections of interconnected software components with complex socio-technical interaction patterns [19, 20]. Typical well-studied ecosystems are software library registries [9, 11, 15] (e.g., npm, PyPI, RubyGems, Maven, Cargo) allowing to reuse software libraries for specific programming languages. Their technical dependency networks grow at a rapid pace and may contain fragile packages that have a high transitive impact [10]. Ecosystem-specific policies, values and technical choices play an important role in how such networks evolve over time [4].

Social issues are at least as important as technical ones. Social coding platforms can lead to effective work coordination strategies [8] and have become indispensable collaborative environments for software ecosystems [13]. Researchers have studied the social aspects of how developer teams interact and evolve [18], how newcomers progress in a software project [25, 26], how the core team grows over time [23], how developer teams get renewed [6], and how socio-technical patterns affect software success or failure [24].

Social and technical issues are tightly interwoven and should be addressed conjointly, because of Conway's law stating that the

ESEC/FSE '19, August 26–30, 2019, Tallinn, Estonia

ACM ISBN 978-1-4503-5572-8/19/08.
https://doi.org/10.1145/3338906.3342497

software structure mimicks the communication and coordination structure of the community developing it [3, 7, 13, 17]. New models are required to better understand such socio-technical congruence at the ecosystem level [22]. In addition, the temporal dimension needs to be taken into account since the contributor and technical relationships evolve over time [5]. Combining social and technical information leads to better prediction models, for example to detect faults in software components [1, 2], to predict if a project participant will become a developer [12], and to recommend experts [16].

3 DATA EXTRACTION AND ANALYSIS

The libraries.io monitoring service contains package metadata, including dependencies for 36 different package managers. I have selected the Cargo package manager for the Rust programming language as a first case study. It was created in 2014, and most of its packages are developed on GitHub. Cargo is growing fast in number of packages, package releases, dependencies and contributors [10].

I used on a datadump of libraries.io [14] to extract the temporal evolution of Cargo's package dependency network. For each package, I extracted the package name, release number and date, maintainer, package dependencies and their versioning constraints. The dataset contained +15K packages, +66K package releases and +48K dependency relations. I retrieved the (optional) link to the corresponding development repositories. I downloaded the relevant historical socio-technical data from their GitHub repositories. This data includes all contributors and their role, the *social* commenting activities they were involved in (+904K comments), and the *technical* development activities they had conducted (+942K commits, +145K pull requests and +266K issues). 3,170 repositories had no comments, and comments follow the Pareto rule (80% of all comments belong to $<$ 20% of all repositories).

To study the socio-technical congruence of Cargo, I am analysing its package dependency network and the associated technical and social activities of its contributors. I report some preliminary anecdotal results below. They need to be complemented with proper statistical hypothesis testing, regression and survival analysis, and prediction modeling. I started to investigate how the presence of commenting activity in a package repository relates to the introduction of dependency to that package. To do so, I considered all packages in which a new dependency was added, and analysed whether commenting activity could be observed in the target package repository *before* or *after* depending on it. Figure 1 summarises the results. One can observe that in more cases commenting activity started after depending on that package. An important shift in behaviour can be observed since September 2016, where the number of packages with commenting activity before starting to depend on them is increasing and even exceeds in September 2017 the number of packages with commenting activity after starting to depend on them. Why this occurs remains an open question for now.

In order to assess which types of comments (i.e., commit comments, issue comments, pull request comments or pull request review comments) are more likely to lead to the introduction of new package dependencies, I analysed their relative proportion in repositories prior to the addition of a dependency toward those packages. Figure 2 presents these results. Among the four types of comments, the proportion of comments on pull requests and issue requests is considerably higher than for commit comments and pull request review comments. I hypothesise that commenting on pull requests and issue requests for a package could serve as a good predictor for adding new dependencies to that package. More detailed statistical analyses are needed to confirm this hypothesis.

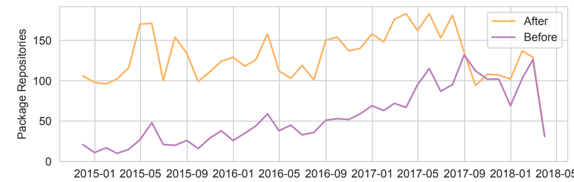

Figure 1: Number of package repositories with first commenting activity *before* or *after* starting to depend on a package.

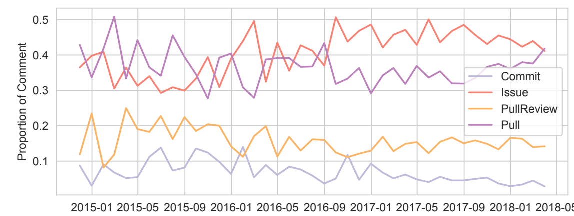

Figure 2: Proportion of comment types made in the repositories of packages before starting to depend on that package.

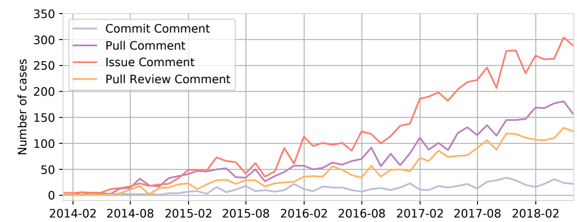

Figure 3: Number of cases with comment activity (by comment type) before starting to technically contribute to a package.

I also started to investigate whether social (i.e., commenting) activity on a package repository increases the likelihood to become technically active on that repository (i.e., submitting commits or pull requests). Figure 3 presents some preliminary results. Considering the four comment types, issue comments appear to be more likely to result in becoming technically active on a package repository. The figure shows the number of observations in which the contributor had a type of comment activity on the package before starting to contribute to it.

4 CONCLUSION

Given that my PhD research project just started, I only have preliminary results about the evolution of the socio-technical congruence of the Cargo packaging ecosystem. During my PhD studies I intend to gain a deeper understanding of the dynamics of this phenomenon, and complement this by studying the expected benefits on the ecosystem's health, such as increased popularity, productivity, responsiveness, contributor intake and retention. I intend to conduct similar studies on other packaging ecosystems, in order to compare their socio-technical congruence and the effect of ecosystem-specific policies and values.

Acknowledgement. This research is supported by the joint FNRS / FWO Excellence of Science project 30446992 SECO-ASSIST.

REFERENCES

[1] P. Bhattacharya, M. Iliofotou, I. Neamtiu, and M. Faloutsos. 2012. Graph-based analysis and prediction for software evolution. In *International Conference on Software Engineering (ICSE)*. 419–429. https://doi.org/10.1109/ICSE.2012.6227173

[2] Christian Bird, Nachiappan Nagappan, Harald Gall, Brendan Murphy, and Premkumar Devanbu. 2009. Putting It All Together: Using Socio-technical Networks to Predict Failures. In *International Symposium on Software Reliability Engineering (ISSRE '09)*. IEEE Computer Society, Washington, DC, USA, 109–119. https://doi.org/10.1109/ISSRE.2009.17

[3] Kelly Blincoe, Francis Harrison, Navpreet Kaur, and Daniela Damian. 2019. Reference Coupling: An exploration of inter-project technical dependencies and their characteristics within large software ecosystems. *Information and Software Technology* 110 (2019), 174 – 189. https://doi.org/10.1016/j.infsof.2019.03.005

[4] Christopher Bogart, Christian Kästner, James Herbsleb, and Ferdian Thung. 2016. How to Break an API: Cost Negotiation and Community Values in Three Software Ecosystems. In *International Symposium on Foundations of Software Engineering (FSE)*. ACM, 109–120. https://doi.org/10.1145/2950290.2950325

[5] Marcelo Cataldo and James D. Herbsleb. 2008. Communication Networks in Geographically Distributed Software Development. In *ACM Conference on Computer Supported Cooperative Work (CSCW '08)*. ACM, New York, NY, USA, 579–588. https://doi.org/10.1145/1460563.1460654

[6] 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

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

[8] Laura A. Dabbish, H. Colleen Stuart, Jason Tsay, and James D. Herbsleb. 2012. Social coding in GitHub: transparency and collaboration in an open software repository. In *Int'l Conf. Computer Supported Cooperative Work*. 1277–1286.

[9] Alexandre Decan, Tom Mens, and Maelick Claes. 2017. An empirical comparison of dependency issues in OSS packaging ecosystems. In *International Conference on Software Analysis, Evolution and Reengineering (SANER)*. 2–12. https://doi.org/10.1109/SANER.2017.7884604

[10] Alexandre Decan, Tom Mens, and Philippe Grosjean. 2019. An empirical comparison of dependency network evolution in seven software packaging ecosystems. *Empirical Software Engineering* 24, 1 (February 2019), 381–416. https://doi.org/10.1007/s10664-017-9589-y

[11] Jens Dietrich, David J. Pearce, Jacob Stringer, Amjed Tahir, and Kelly Blincoe. 2019. Dependency Versioning in the Wild. In *International Conference on Mining Software Repositories (MSR)*.

[12] Mohammad Gharehyazie, Daryl Posnett, and Vladimir Filkov. 2013. Social Activities Rival Patch Submission For Prediction of Developer Initiation in OSS Projects. In *Int'l Conf. Software Maintenance*.

[13] J. D. Herbsleb and R. E. Grinter. 1999. Architectures, coordination, and distance: Conway's law and beyond. *IEEE Software* 16, 5 (1999), 63–70.

[14] Jeremy Katz. 2018. Libraries.io Open Source Repository and Dependency Metadata (Version 1.4.0) [Data set]. http://doi.org/10.5281/zenodo.2536573.

[15] R. Kikas, G. Gousios, M. Dumas, and D. Pfahl. 2017. Structure and Evolution of Package Dependency Networks. In *International Conference on Mining Software Repositories (MSR)*. 102–112. https://doi.org/10.1109/MSR.2017.55

[16] 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

[17] I. Kwan, A. Schroter, and D. Damian. 2011. Does Socio-Technical Congruence Have an Effect on Software Build Success? A Study of Coordination in a Software Project. *IEEE Trans. Soft. Eng.* 37, 3 (May 2011), 307–324. https://doi.org/10.1109/TSE.2011.29

[18] 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

[19] Mircea Lungu. 2009. *Reverse Engineering Software Ecosystems*. Ph.D. Dissertation. University of Lugano.

[20] Konstantinos Manikas and Klaus Marius Hansen. 2013. Software Ecosystems: A Systematic Literature Review. *J. Systems and Software* 86, 5 (May 2013), 1294–1306. http://dx.doi.org/10.1016/j.jss.2012.12.026

[21] Tom Mens, Marcelo Cataldo, and Daniela Damian. 2019. The Social Developer: The Future of Software Development. *IEEE Software* 36 (January–February 2019). https://doi.org/10.1109/MS.2018.2874316

[22] 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

[23] 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.

[24] 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*.

[25] 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

[26] 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.