Towards solving social and technical problems in open source software ecosystems

Using cause-and-effect analysis to disentangle the causes of complex problems

Josianne Marsan, Mathieu Templier and Patrick Marois, *Université Laval*

Bram Adams, *Polytechnique Montréal*

Kevin Carillo, *Toulouse Business School*

Georgia Leïda Mopenza, *Université Laval*

Managing large-scale development projects in open source software ecosystems involves dealing with an array of technical and social problems. To disentangle the causes of such problems, we interviewed experts and performed a cause-and-effect analysis. Our findings demonstrate that loss of contributors is the most important social problem, while poor code quality is the most important technical problem, and that both problems result from complex socio-technical interrelations of causes. Our approach suggests that cause-and-effect analysis can help to better understand problems in open source software ecosystems.

Keywords: open source software; software ecosystem; cause-and-effect analysis; code quality; loss of contributors

Open source software (OSS) development is key to the growth of the software industry, to the extent that many OSS today are leaders in their markets[1]. OSS projects increasingly tend to regroup into large-scale projects or "software ecosystems" to reduce effort and accelerate innovation[2]. A software ecosystem (e.g., Android, Debian) is a set of businesses interacting with the same market for software and services, and their exchanges of artefacts, resources, and information[3].

The "health" of a software ecosystem refers to the normal functioning of its constituting projects, as well as the ecosystem as a whole[4]. Health problems tend to propagate throughout the ecosystem by way of technical or social dependencies[5]. Thus, managing the health of OSS ecosystems (OSSECOs) is challenging[6], since it is natural for OSS practitioners to focus on technical rather than social aspects. Yet, in OSSECOs, continuous collaboration between various stakeholders[7] is required.

Managing OSSECO health therefore requires a holistic view of complex socio-technical aspects. Social and technical problems are dependent events, usually linked through cause-effect relations. Hence, studying them requires powerful approaches able to model such relations.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MS.2018.2874323, IEEE Software

Since only a few tools and techniques exist to address OSSECOs health problems[8], we use an original approach, consisting of interviews with experts, Content Analysis and Root Cause Analysis (RCA). Our findings show that loss of contributors and poor code quality are important problems that result from complex socio-technical interrelations of causes.

Analysis of expert opinions

Three of the authors interviewed ten OSSECO experts during the 2017 Open Source Summit (OSSummit) in Prague. One author validated the results of our analysis by interviewing an additional expert. Each interviewee had to meet at least one criteria of inclusion (see Table 1).

Table 1. Profile of experts.

Expert number	*Inclusion criteria*	*Role in OSSECOs*	*OSS expertise*	*OSS experience (in years)*	*OSS status*	*Current company size*	*Current company sector*	*Current country*
Experts interviewed to collect their opinion on health problems of SECO and their causes								
1	C1; C3; C4	D	T	10	P; H	S	Analytics for OSS development monitoring	Spain
2	C1; C4	D; M	T	23	P; H	XS	Linux support and consulting	Germany
3	C1; C4	D; M	T	17	P; H	XS	Embedded Linux design and development	Finland
4	C1; C4	CM	S	10	P; H	S	Open source enterprise solutions	Italy
5	C1; C4	CM	S	8	P; H	L	Open source enterprise solutions	Finland
6	C1	D; SP	T	20	P; H	XL	Cloud and virtualization software and	USA

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MS.2018.2874323, IEEE Software

							services	
7	C1	D; SP	T	8	P; H	S	Open source big data solutions	France
8	C1	D; M	T	17	P; H	S	Open source software and hardware consulting	Canada
9	C1	D	T	9	P; H	S	OSS development, licensing and training	Canada
10	C1	D	T	4	P	XL	Semiconductors and integrated circuits	USA
Expert interviewed to validate the analysis of the experts' opinions done by the authors (see figure 2)								
11	C1; C4	D; M	T	20	H	L	Education and research	Canada

Note: *Our sample represents high experience and diversity, but the representation of experts of the C1 or C4 type is higher than the representation of experts of the C2 or C3 type. This "imbalance" in the expert profiles may imply that the results are under representative and hence provide only a partial view of the reality.*

LEGEND

Inclusion criteria :

C1: Being a contributor (e.g., Community Manager, Developer, Documentation Producer, Service Provider) to an OSSECO, or having been one in the last five years;

C2: Holding a management position in an OSS foundation (e.g., Linux Foundation, Apache Software Foundation), or having held one in the last five years;

C3: Being involved in the development of at least one software application aimed at improving the quality of development of software projects or ecosystems (e.g., Bitergia's GrimoireLab, SonarQube by SonarSource, DependencyCI), or having been involved in the last five years;

C4: Being a guest speaker at major events on OSSECOs (e.g., OSSummit, OSCON), or having been one in the last five years.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MS.2018.2874323, IEEE Software

Role in OSSECOs:	**OSS expertise:**	**OSS status:**	**Company size (number of employees) :**
D: Developer;	T: Technical;	P: On payroll;	XS: 1-9;
M: Maintainer;	S: Social.	H: Hobbyist.	S: 10 – 249;
CM: Community Manager;			M: 250 – 4999;
SP: Service Provider.			L: 5000 – 14999;
			XL: 15000+.

We conducted semi-directed individual interviews[9] with the first ten experts (mean duration of 47.5 minutes). Experts answered freely to the question “According to your experience, what are the most important health problems that OSSECOs encounter and what causes each of these problems?”. Further, we interviewed the eleventh expert to validate Figure 2.

Cause-and-effect analysis

Figure 1 presents our analysis process and its outputs.

The first step, Content Analysis, has been used in software engineering [(e.g., 10)], as it allows to make inferences of causal relationships thanks to categories guiding the coding of interviews[11]. Our categories reflected core concepts in RCA, a useful technique to uncover cause-and-effect chains[12]. Then, we performed RCA by consolidating our content analysis results into cause-and-effect diagrams for two problems. Finally, we merged them into a single diagram showing the most frequent chains (see Figure 2). Descriptive frequencies reflect the importance of themes[10] for our group of experts.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MS.2018.2874323, IEEE Software

Figure 1. Cause-and-effect analysis process and outputs.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MS.2018.2874323, IEEE Software

Evidence "from the trenches"

This section discusses the health problems, the causal chains, and the actionable causes.

Health problems

Important (i.e., mentioned by at least 50% of experts) social problems are, in order of importance:

1. Loss of contributors
2. Lack of interactions between contributors
3. Low number of contributors
4. Divergence of interests/directions in the OSSECO

Important technical problems are:

1. Poor code quality
2. Low number of code commits

The top social problem is "Loss of contributors", mentioned by all experts, while the top technical problem is "Poor code quality", mentioned by 70%. These are the only problems mentioned by more than 50%. The literature has also established both problems as being important to practitioners [e.g., 13-15].

Cause-and-effect chains

Figure 2 shows the cause-and-effect chains that resulted from our analysis of "Loss of contributors" and "Poor code quality". Our results suggest that managing these problems requires a particular attention to three properties of cause-and-effect chains, discussed next.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MS.2018.2874323, IEEE Software

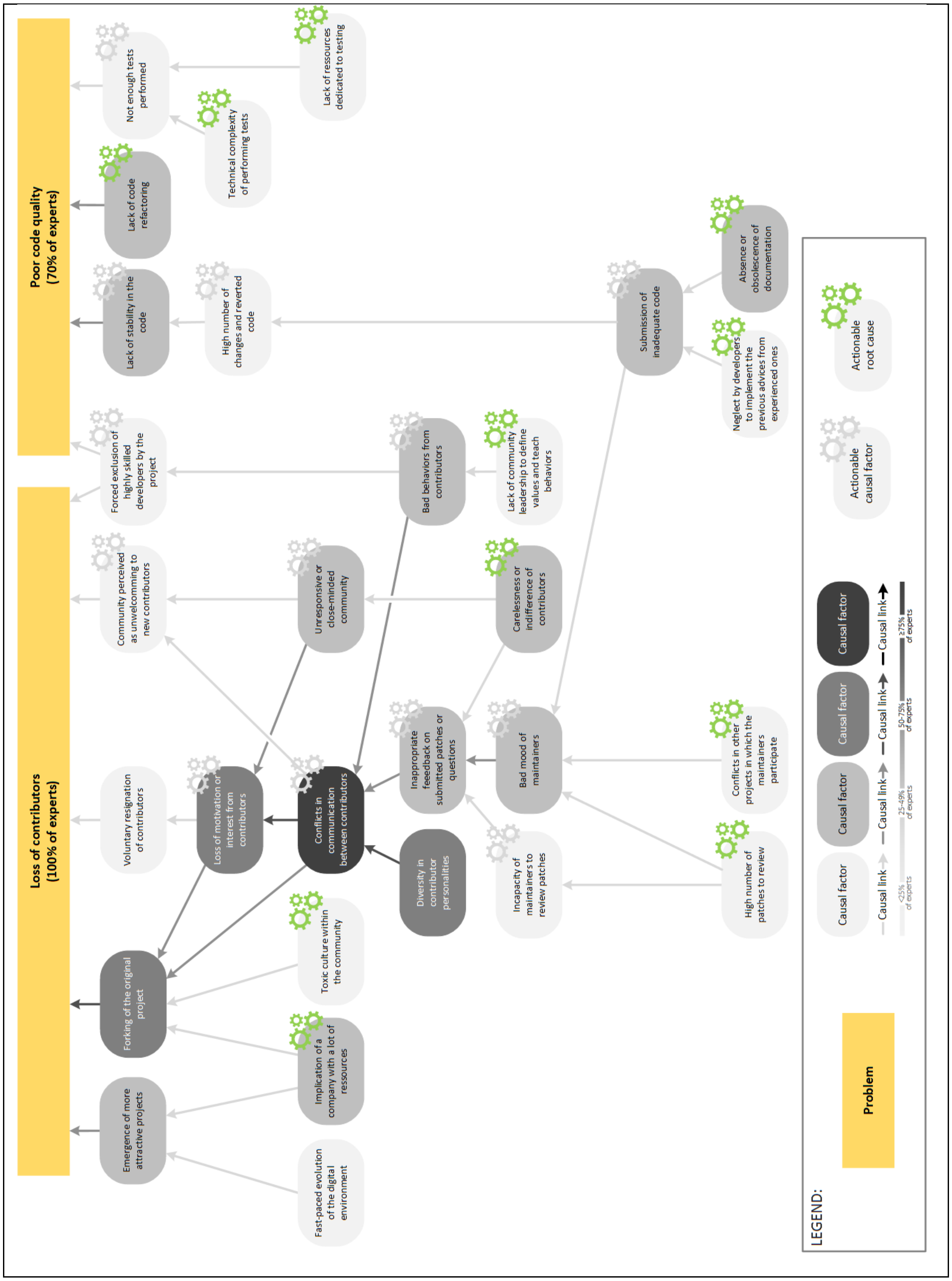


Figure 2. Cause-and-effect diagram.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MS.2018.2874323, IEEE Software

Transversality of chains. Our pool of experts mentioned that the health of the projects making up an OSSECO mostly reflects the health of this OSSECO. This originates from the dependencies and interconnectedness between OSSECO components. Our data confirms this assertion and demonstrates that the chains can be transversal, i.e., crossing multiple projects.

Projects within an OSSECO can collaborate but also compete: *"some other projects might start being the one [,...] going new places, doing cool work, and you're sort of the backwater project"* (Expert_6). According to experts, explanations as to why contributors perceive projects as more attractive are for example their focus on trendy technologies or the investment of important resources from companies. When contributors get more interested in another project outside of the OSSECO, they become likely to leave.

Another example from Figure 2 is maintainers contributing in many projects and carrying stress and conflicts from one project to the other. Bad mood might lead them to provide harsh feedback, thus leading to the development of conflicts in communication between contributors. Hurt developers may then lose interest in the project.

Multi-roleness of chains. OSSECOs involve the collaboration and interaction between multiple stakeholders[7]. Indeed, our analysis demonstrates that cause-and-effect chains can cover multiple roles within an OSSECO. For instance, Figure 2 involves maintainers, new and experienced developers, community managers, users, and companies.

Figure 2 emphasizes two examples of multi-roleness. First, diversity in contributor personalities might lead to conflicts in communication. When reviewing patches or answering questions, contributors "*may just be coming across as very harsh or authoritative people*" (Expert_5) or "*lose patience or lose [their] temper*" (Expert_2). Second, some communities that are less inclusive or open-minded, are unwelcoming to new contributors, who might not "*feel safe to contribute, feel safe to ask question*" (Expert_4) and, as a result, are likely to leave the project.

Socio-technicality of chains. Problems are rooted in the intertwining of a number of technical causes (e.g., submission of inadequate code) and social causes (e.g., bad behaviors from contributors). Our data uncovers this entanglement and shows that the cause-and-effect chains of OSSECO health problems are indeed socio-technical. Interestingly, the majority of causal factors and problems in our study are of a social nature, supporting the idea that OSSECO contributors' work and practices shift from a purely technical to a more socio-technical perspective.

The submission of inadequate code, a technical cause that belongs to the cause-and-effect chains of both focal problems, has two potential effects according to our experts. On the technical side, inadequate code might lead to a high number of changes and reverted code: "*Maybe there was a lot of turn [...] and [the code] was reverted. They went back to it a few days later with a few changes and then some changes, and then some changes, and then some changes"* (Expert_6). The submission of inadequate code might also trigger frustration from maintainers and lead to social issues involving tensions between actors: "*the basic things [are to] try to write clear code, try to comment the code. [...] You can have some tension in the project, because someone has to review that code*" (Expert_1).

Another interesting example of the socio-technical sources of problems refers to the technical tests, which, when insufficient, might hinder the quality of the code. As shown in Figure 2, explanations as to why under-testing exists include technical complexity of performing tests (e.g., "[testing for] *backward compatibility is far, far, far from being evident*" (Expert_7)) and the lack of resources dedicated to testing (e.g., "*Do we really take the time, write the tests, and capitalize on this for the future? But this will take a lot of time, so at the end we'll fix it very quickly.*" (Expert_7)).

Actionable causes

As shown in Figure 2, not all causes are actionable, i.e., it is not possible to eradicate or minimize each cause (e.g., the fast-paced evolution of the digital environment). Root causes are the ones that we want to act upon first, as addressing these deepest causal factors will most likely provide a more sustainable relief. For instance, bad behaviors from contributors leads to the important issue of conflicts in communication, i.e., the "*kind of things that can basically kill communities*" (Expert_1). Sanctioning improper conduct is a possible solution, but addressing the lack of community leadership, for instance by appointing a "*community manager [in charge] of teaching the community how to behave*" (Expert_1) appears as a more long-term resolving strategy.

Similarly, pivotal causes, i.e., causal factors that are sources of multiple branches in the chains, will potentially have far-reaching effects. For example, the experts mentioned that carelessness or indifference of contributors, notably when companies pay developers to write code for the project, might trigger two other causal factors. Compared to hobbyists, they are often less passionate about the project. This might then influence developers' actions on mailing lists, where they could become "*not open to discuss, [...] to change [their] idea*" (Expert_4), or shape the whole community toward being less open and less responsive. Solutions for this issue are not easy, nor straightforward. Community managers could identify the companies to inform them directly about their problematic developers, or they could implement motivational programs within the OSSECO to encourage developers' excellent work.

This paper adds to the volume of work on "Loss of contributors" and "Poor code quality" by performing cause-and-effect analysis on these problems in the context of OSSECOs. Our findings emphasize three important properties of cause-and-effect chains. We hope our work will ignite in OSSECOs professionals a proactive mindset toward OSSECO health. Indeed, in order to prevent the occurrence of two serious problems in OSSECOs, it is important to treat the actionable causes uncovered, even if they may seem innocuous in themselves. Professionals must consider chains of causes since we uncovered chains as deep as four levels of causes for "Poor code quality" and seven for "Loss of contributors".

Acknowledgments

We are thankful to the OSSECO experts that invested their time in this study and to the Editors and three anonymous reviewers for their helpful comments and suggestions. We acknowledge support from FRQ-FNRS grant 264544.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MS.2018.2874323, IEEE Software

References

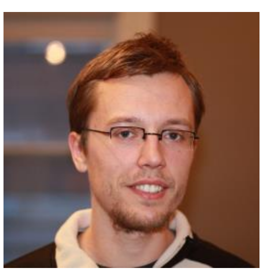
1. R. Schuwer, M. van Genuchten, L. Hatton. On the impact of being open. IEEE Software 32(5), 2015, 81-83.
2. J. Herbsleb, C. Kästner, C. Bogart. Intelligently Transparent Software Ecosystems. IEEE Software 33(1), 2016, 89-96.
3. S. Jansen, A. Finkelstein, S. Brinkkemper. A sense of community: A research agenda for software ecosystems. International Conference on Software Engineering, 2009, 187-190.
4. S. Jansen. Measuring the health of open source software ecosystems: Beyond the scope of project health. Information and Software Technology, 56(11), 2014, 1508-1519.
5. K. Carillo, J. Marsan, B. Negoita. Exploring the biosphere - Towards a conceptualization of peer production communities as living organisms. AIS SIGOPEN Developmental Workshop for Research on Open Phenomena, 2017.
6. T. Mens. An ecosystemic and socio-technical view on software maintenance and evolution. International Conference of Software Maintenance and Evolution, 2016, 1-8.
7. P.B. de Laat. Governance of open source software: state of the art. Journal of Management & Governance, 11(2), 2007, 165–177.
8. R. Kazman, H.M. Chen. The metropolis model and its implications for the engineering of software ecosystems. Workshop on Future of software engineering research, 2010, 187-190.
9. M.B. Miles, A.M. Huberman. Qualitative Data Analysis (Second Edition), Sage Publications, 1994.
10. S. Beecham, T. Hall, A. Rainer, A. Software process improvement problems in twelve software companies: an empirical analysis. Empirical Software Engineering, 8(1), 2003, 7-42.
11. A. Coners, B. Matthies. A content analysis of content analyses in IS research: purposes, data sources, and methodological characteristics. Pacific Asia Conference on Information Systems, 2014, 111.
12. J.J. Rooney, L.N.V. Heuvel. Root cause analysis for beginners. Quality progress, 37(7), 2004, 45-56.
13. I. Stamelos, L. Angelis, A. Oikonomou, G.L. Bleris. Code quality analysis in open source software development. Information Systems Journal, 12(1), 2002, 43–60.
14. D. Izquierdo-Cortazar, G. Robles, F. Ortega, J.M. Gonzalez-Barahona. Using Software Archaeology to Measure Knowledge Loss in Software Projects Due to Developer Turnover. Hawaii International Conference on System Sciences, 2009, 498.
15. K. Carillo, S. Huff, B. Chawner. What makes a good contributor? Understanding contributor behavior within large Free/Open Source Software projects – A socialization perspective. The Journal of Strategic Information Systems, 26(4), 2017, 322–359.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MS.2018.2874323, IEEE Software

Josianne Marsan (PhD, HEC Montréal) is Professor in Information Systems (IS) at Université Laval (Canada). She is a member of the Board of Governance of the CHAOSS project, and co-investigator of the SECOHealth project (https://secohealth.github.io/). Contact her at josianne.marsan@sio.ulaval.ca.

Mathieu Templier (PhD, HEC Montréal) is Assistant professor in IS at Université Laval. He is a collaborator in SECOHealth. Contact him at mathieu.templier@sio.ulaval.ca.

Patrick Marois is a PhD candidate at Université Laval. He is a member of SECOHealth. Contact him at patrick.marois.1@ulaval.ca.



Bram Adams (PhD, Ghent University) is Associate professor in software engineering at Polytechnique Montreal (Canada). He is head of the Lab on Maintenance, Construction and Intelligence of Software, and a principal investigator in SECOHealth. Contact him at bram.adams@polymtl.ca.

Kevin Carillo (PhD, Victoria University of Wellington) is Associate professor in IS at Toulouse Business School (France). He is a collaborator in SECOHealth. Contact him at k.carillo@tbs-education.fr.

Georgia Leïda Mopenza is a M.Sc. student at Université Laval. Contact her at georgia-leida.mopenza.1@ulaval.ca.

Summary of our question and findings

1. “From the trenches” cause-and-effect analysis of social or technical problems in open source software ecosystems
2. Loss of contributors is the most important social problem in open source software ecosystems according to experts
3. Poor code quality is the most important technical problem in open source software ecosystems according to experts
4. Main health problems of open source software ecosystems result from complex socio-technical chains of causes
5. Cause-and-effect analysis can help to better understand problems in open source software ecosystems