Community-based software development for MDE tools

Jordi Cabot[1,2], Javier Luis Cánovas Izquierdo[2], Valerio Cosentino[2]

[1] ICREA
Barcelona, Spain
jordi.cabot@icrea.cat
[2] UOC
Barcelona, Spain
{jcanovasi,vcosentino}@uoc.edu

Abstract. Many open source projects stagnate after an initial push and end-up fading away. In this talk we will argue that, most of the time, the reason has nothing to do with the quality of the software itself but with the project's inability to attract and support a healthy community around it. The community of contributors and also the users must take an active role. MDE tools are not an exception to this challenge. We will review several actions and strategies that OSS project managers of MDE tools could put into practice to reverse this situation, mostly taken from other disciplines like social science, economy and political science.

1 Introduction

Effective collaboration requires adequate technical solutions, but they alone are not enough. Adoption of good organizational practices and development processes within the development team is also a must. This is especially true when we have distributed teams as it is typically the case in most complex software projects nowadays.

Getting these collaboration models to work at a large scale is very challenging. That is why many software projects decide to embrace the principles of Open Source Software. According to the Open Source Initiative[1]: *"OSS development is a development method that harnesses the power of distributed peer review and transparency*". OSS is typically developed in a collaborative manner via online code hosting platforms like GitHub. This is the main difference with respect to proprietary software: not only the code is open (free for everybody to access and modify) but also the development is (supposedly) performed in the open which favours the collaboration of the whole community behind the software, including its users. This active participation of end users in the development process is already defended all agile methodologies.

This distinction was clearly illustrated in the well-known essay, and later a book, "The Cathedral and the Bazaar" [1] based on the author's observations of the

[1] https://opensource.org/

development of Linux kernel that the author tested and validated on his own open source project later on. This essay contrasts two development models: the *Cathedral* model where code is developed by a restricted set of developers and the *Bazaar* model where development is a collaborative endeavor and users are co-developers allowing for rapid code improvement, effective debugging and aligned software evolution. This "co-developer" role does not mean users contribute code, it highlights the fact that users are key members of the software community, have a say in it and can contribute in any form or shape they are able to, e.g. submitting bug reports, feature requests or just giving feedback on any aspect of the software.

Nevertheless, in practice, many OSS projects are not as open as they should be. Yes, their source code is freely accessible but the management of the project and its leadership is not transparent and hardly ever follows any kind of democratic practices, making it extremely difficult for users and the community in general to influence the development of the project. For instance, we manually analyzed the twenty-five most popular projects in GitHub and found out that only one (4%) explicitly described how user contributions would be managed, with another 28% giving partial hints. This means that 68% had no explicit governance model[2]. Absolutely none of them were democratic (i.e. end users could not vote in any way not even to elect people to represent them). This situation is similar in other open source platforms. For instance, in Eclipse they even have established an award to the most open project to *"recognize the open source project that best exemplifies the openness, transparency and diversity expected of great open source projects"*[3].

And this is not the only problem. Most projects struggle to attract contributors. In fact, more than two thirds of all projects in GitHub have only one or two contributors [2], which of course limits a lot of the interaction they may have with the users of that software specially if they have a large user base. Therefore, we can conclude the OSS model, as it is now, is broken. This affects the long-term sustainability of the projects, with many projects failing and getting abandoned in the very early stages, see [3] for some statistics.

This is especially relevant for the MDE community which, as a "new" tool ecosystem, has largely depended on community contributions to develop and industrialize [4] all kinds of modeling tools.

In this paper we will explore the main challenges of OSS development from a community perspective and how we believe a more community-based software development process could help overcome them.

2 State of the art

Open source software development has been widely studied and from several different points of view. Our literature review on works studying GitHub OSS projects resulted in over 100 papers to classify [5].

[2] A governance model describes the roles that project participants can take on and the process for decision making within the project (OSS watch)

[3] https://bugs.eclipse.org/bugs/show_bug.cgi?id=484321

Most papers analyze software projects from a code-centric perspective meaning that their focus is the analysis of the projects' source code by evaluating, for instance, (1) the use of programming languages, (2) the type of license they apply, (3) the folder structure of the project, (4) the technological domain, or (5) the potential vulnerabilities and complexity of the code.

Instead, only a few works analyze the social part of the software development process, trying to understand how developers are internally organized and work together in the project. Some studies aim at providing a static view of the organization behind OSS projects, in particular, to analyze: (1) the different kinds of communities (e.g. [6]), (2) its structure (e.g.[7], [8][2][9]), (3) its diversity (e.g. [10], [11], [12]), (4) the profile of the users in the community (e.g. [13], [14]). or (5) their popularity (e.g. [15], [16]). Other works focus more on the community dynamics by analyzing the interactions between community members and the project or among members themselves. such as the social ties and patterns among contributors (e.g.[17], [18]), the social and technical factors that motivate people to contribute to a given project (e.g. [19]) and assignment algorithms that recommend developers to open tasks (e.g. [20]).

Open Challenges. Based on this literature review we can conclude that (open-source) software development faces the following challenges:

- It is not as open as you would expect (the code is open, the management and decision-making of the project is not) which leads to a number of governance, decision and leadership problems,
- it has strong difficulties in attracting new contributors (specially technical ones) to enlarge the project community with the most appropriate profiles to get the project advance fast (including, for instance, both improvements / additions at the code level and decisions on issues and bugs prioritization for future releases),
- it is unable to manage its community efficiently, integrating all different profiles (owners, contributors, users,...) in a way they all can smoothly interact and collaborate to make the project evolve aligned with their needs,

which hampers people's experience with open source thus threatening the project's sustainability and future success.

We believe these problems are shared across all domains of OSS and therefore MDE is not an exception, though there is a lack of specific studies on MDE OSS tools. In the following we provide some ideas to tackle these issues.

3 Community-based software development: a Roadmap

We believe any significant gain in OSS development implies shifting our main focus of attention from the analysis of code aspects in the software repository to the analysis of the people behind that code, either as developers, owners or users.

The following figure illustrates this change of perspective, highlighting how we go from the current developer centric view (kind of a meritocracy where only core developers have the right to decide) to a community that now collaborates together and has the tools it needs to manage this collaboration. The left-hand side summarizes our perception of the current situation where users, either technical users contributing new patches of code or non-technical users just reporting bugs or asking for new features, have no real influence on the core project management team that lives in a bubble. The right-hand side shows what happens when this isolation "bubble" is burst and everybody has the chance to have an active (and decisive) participation. From this point on, we can say that there is a real community behind the project. And this community is not alone either but it is part of a larger ecosystem of OSS communities that can collaborate with each other for further global optimizations.

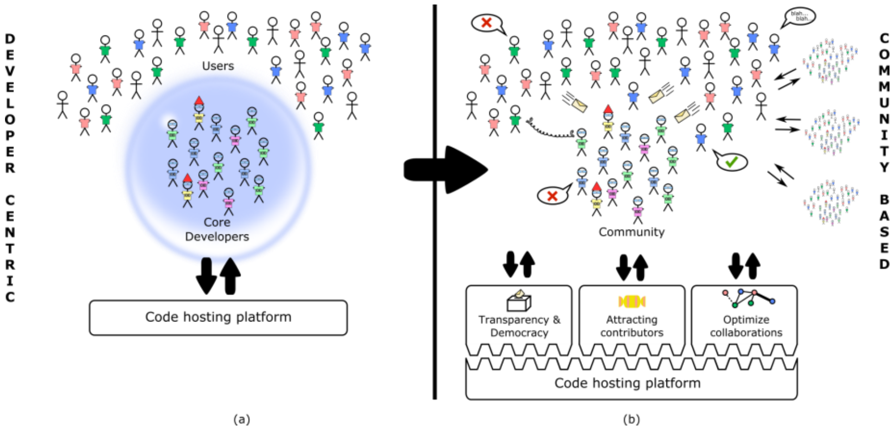

Fig. 1. **Developer-centric vs Community-based software development**

This community-driven process will be enabled by borrowing and **adapting** to the software development field techniques from the domains of **political science**, **sociology** (e.g. social/behavioural informatics), **economics** and **ecology** that have been studying a diverse range of communities for centuries. Among all of them, we would like to highlight three main open research areas of interest (Fig. 1 b) we believe specially important:

A1. **Bring Transparency and Democracy to OSS development,** by facilitating projects to open up all aspects of the project (and not only its source code) via the definition of a precise governance model setting up the foundations of a participative process. Open source communities are not as open as they seem as discussed above and their governance is an open research challenge [21]. As an example, their lack of transparency and frequent anti-democratic decision procedures can scare away potential contributors/users[4] and hamper its alignment with the needs of the community. At the very least, a clear definition of the project's governance model would be necessary (e.g. using this DSL [22]). The adaptation of different kinds of political systems (including different types of democratic models, e.g. representative,

[4] Even if, for whatever reason, a certain project is NOT looking for contributors, stating this clearly (transparency) would avoid misunderstandings.

participative, liquid,...) to the specific context of OSS and helping projects transition their internal organization to one of these systems if they wish so is work in progress.

A.2. Increase the number of contributors to OSS projects, by providing projects with innovative tools to attract new contributors and favor their long-term involvement in the project. OSS projects need contributors to progress [13][23]. New retribution models (inspired from the study of market economy, in particular matching markets where money is not the main factor [25]) to convince more people to join, gamification strategies to increase their participation and profiling tools to help project owners identify and reach out potential candidates (to cover technical gaps or increase diversity) cross-profiling the project needs with online profiles in social networks are necessary.

A.3. Optimize internal project collaborations by giving to project owners the tools they need to identify collaboration bottlenecks and to individual users the motivation and information they need to be more effective. Effective collaboration requires more than setting up theoretical good conditions for it. A continuous monitoring of the community structure and exchanges taking place would allow detecting early on possible misconfigurations in the community giving the opportunity to act on them as soon as possible. Most of the possible issues can be detected by representing the community as a directed multigraph and adapting graph algorithms to collect some metrics on it (connected components, low density areas, key nodes,...). These metrics will then be compared with a range of acceptable values derived from a benchmark of "successful" projects to identify worrisome deviations.

References

[1] E. S. Raymond, *The Cathedral and the Bazaar*. O'Reilly Media, 2001.

[2] E. Kalliamvakou, G. Gousios, K. Blincoe, L. Singer, D. M. German, and D. Damian, "An in-depth study of the promises and perils of mining GitHub," *Empirical Software Engineering*, Sep. 2015.

[3] C. M. Schweik and R. C. English, *Internet Success: A Study of Open-Source Software Commons*. The MIT Press, 2012.

[4] H. Brunelière and J. Cabot, "On Developing Open Source MDE Tools: Our Eclipse Stories and Lessons Learned," in *OSS4MDE@MoDELS 2014*, 2014, pp. 9–19.

[5] V. Cosentino, J. L. C. Izquierdo, and J. Cabot, "Findings from GitHub: methods, datasets and limitations," in *Proceedings of the 13th International Conference on Mining Software Repositories, {MSR} 2016, Austin, TX, USA, May 14-22, 2016*, 2016, pp. 137–141.

[6] B. Vasilescu, V. Filkov, and A. Serebrenik, "Perceptions of Diversity on GitHub : A User Survey," in *CHASE Workshop*, 2015.

[7] N. Matragkas, J. R. Williams, D. S. Kolovos, and R. F. Paige, "Analysing the 'biodiversity' of open source ecosystems: the GitHub case," in *MSR conf.*, 2014, pp. 356–359.

[8] A. E. Hassan and S. Mcintosh, "Revisiting the Applicability of the Pareto Principle to Core Development Teams in Open Source Software Projects Categories and Subject

Descriptors," in *IWPSE workshop*, 2015, pp. 46–55.

[9] M. Y. Allaho and W.-C. Lee, "Trends and behavior of developers in open collaborative software projects," in *2014 International Conference on Behavioral, Economic, and Socio-Cultural Computing (BESC2014)*, 2014, pp. 1–7.

[10] A. Lima, L. Rossi, and M. Musolesi, "Coding together at scale: GitHub as a collaborative social network," in *8th AAAI International Conference on Weblogs and Social Media*, 2014, pp. 295–304.

[11] B. Vasilescu, V. Filkov, and A. Serebrenik, "Perceptions of Diversity on GitHub : A User Survey," *CHASE Workshop*, 2015.

[12] D. N. Beede, T. A. Julian, D. Langdon, G. McKittrick, B. Khan, and M. E. Doms, "Women in STEM: A Gender Gap to Innovation," *SSRN Electronic Journal*, Aug. 2011.

[13] L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb, "Social coding in github: transparency and collaboration in an open software repository," in *15th ACM Conference on Computer Supported Cooperative Work*, 2012, pp. 1277–1286.

[14] L. Dabbish, C. Stuart, J. Tsay, and J. Herbsleb, "Leveraging Transparency," *IEEE Software*, vol. 30, no. 1, pp. 37–43, Jan. 2013.

[15] K. Aggarwal, A. Hindle, and E. Stroulia, "Co-evolution of project documentation and popularity within github," in *MSR conf.*, 2014, pp. 360–363.

[16] S. Weber and J. Luo, "What Makes an Open Source Code Popular on Git Hub?," in *ICDMW conf.*, 2014, pp. 851–855.

[17] M. Y. Allaho and W.-C. Lee, "Analyzing the Social Ties and Structure of Contributors in Open Source Software Community," in *ASONAM conf.*, 2013, pp. 56–60.

[18] Y. Yu, G. Yin, H. Wang, and T. Wang, "Exploring the Patterns of Social Behavior in GitHub," in *CrowdSoft Workshop*, 2014, pp. 31–36.

[19] J. Tsay, L. Dabbish, and J. Herbsleb, "Influence of social and technical factors for evaluating contribution in GitHub," in *36th International Conference on Software Engineering*, 2014, pp. 356–366.

[20] J. Xavier and A. Macedo, "Understanding the popularity of reporters and assignees in the Github," in *26th International Conference on Software Engineering and Knowledge Engineering*, 2014, pp. 484–489.

[21] A. Serebrenik and T. Mens, "Challenges in Software Ecosystems Research," in *Proceedings of the 2015 European Conference on Software Architecture Workshops - ECSAW '15*, 2015, pp. 1–6.

[22] J. L. Canovas Izquierdo and J. Cabot, "Enabling the Definition and Enforcement of Governance Rules in Open Source Systems," in *ICSE SEIS conf.*, 2015, vol. 2, pp. 505–514.

[23] R. Padhye, S. Mani, and V. S. Sinha, "A study of external community contribution to open-source projects on GitHub," in *Proceedings of the 11th Working Conference on Mining Software Repositories - MSR 2014*, 2014, pp. 332–335.

[24] J. L. Cánovas Izquierdo, V. Cosentino, and J. Cabot, "Popularity will NOT bring more contributions to your OSS project," *Journal of Object Technology*, vol. 14, no. 4, 2015.

[25] A. E. Roth, *Who Gets What — and Why: The New Economics of Matchmaking and Market Design*. Eamon Dolan/Houghton Mifflin Harcourt, 2015.