Collaborative Modeling in Open Source Systems

[1]Omar Badreddin, [2]Wahab Hamou-Lhadj, [3]Vahdat Abdelzad,
[4]Rahad Khandoker, [5]Maged Elassar

[1]University of Texas, El Paso, TX, USA
obbadreddin@utep.edu
[2]Concordia University, Montreal, QC, Canada
wahab.hamou-lhadj@ concordia.ca
[3]Waterloo University, Waterloo, ON, Canada
vahdat.abdelzad@uwaterloo.ca
[4]University of Texas, El Paso, TX, USA
rahad.baten@yahoo.com
[5]NASA Jet Propulsion laboratory, Pasadena, CA, USA
maged.e.elaasar@jpl.nasa.gov

Abstract—The Open source ecosystem creates new pathways for participation and collaboration from a broad and diverse community of developers. As a software system grows, the need to capture its design, often through models, becomes important to boost communication and collaboration. In this paper, we report on a study that assesses the open source community's adoption of modeling as a way to capture design and enable collaboration among development teams. The study includes a search of open source repositories looking for modeling artifacts, a survey, a questionnaire, and a set of interviews with open source contributors. Our findings show that there is a low number of modeling artifacts that are included in open source project repositories. However, the survey, the questionnaire, and the interviews suggest that capturing design in models is much more common than what can be inferred by searching the repositories alone. These models are created through collaborations, but are not necessarily shared in the open source repositories. This is due to many factors including the lack of incentives to share beyond the immediate circle of collaborators.

Keywords: Model Driven Software Development, Open Source, Collaborative Modeling, Empirical Investigation.

1 Introduction

Open source software (OSS) has demonstrated numerous successes in supporting large-scale collaborative projects. OSS is unique in its support for collaborative development because of its inert ability to attract and sustain a community of users and developers. It is common for an OSS project to include hundreds of developers contributing to the same project.

However, many OSS projects are developed with little structure, heavily relying on the vigilance of contributors and a few champions. Adoption of UML and other design languages is particularly scarce 1. This lack of structured development means

that OSS often accumulates significant technical debts and suffers from unnecessary and avoidable code complexities 14. This, in turn, obscures the knowledge embedded in the algorithms and codes, limits reuse, and makes code prohibitively expensive to maintain, upgrade, scale, or extend.

Investigations of open source modeling practices often focus on mining artifacts from open source repositories 10. These studies provide valuable insights into the types and nature of modeling practices used in open source projects. However, such studies are limited in their scope, as some artifacts are not published as part of an open source project. In this paper, we conduct a study that not only investigates open source repositories, but also includes extensive input from open source contributors. Specifically, our study 1) searches repositories looking for modeling artifacts, 2) surveys open source contributors, and 3) collects data from questionnaires and interviews to gain further insights into the practice. This paper extends our previous work on surveying software engineering practitioners 9 and investigating open source development practices 1, including open source collaborative design 4.

There are a few studies that focus on collaborative modeling in open source environments. Sack et al. 7 described a methodological framework that combines ethnography, text mining, and socio-technical network analysis and visualization to understand OSS development in its totality. Ho-Quang et al. 3 analyzed open source projects for evidence of modeling. While they found that modeling activities are rather scarce in open source, those who do adopt them report increased productivity and code quality. Low adaption of modeling practices particularly in open source projects has also been reported by other studies 28. Nakagawa presented a case study that established the relationship between software architecture and code quality in open source 12. Gaar and Teiniker 13 analyzed model-based design collaboration in open source, and demonstrated the potential for using social media platforms to facilitate global model-based collaboration.

2 Study Design

Our study includes 1) investigation of open source artifacts, 2) a survey targeting open source contributors and users, and 3) a questionnaire and interview study with open source contributors and users. The aim of the study is to answer the following questions:

- **RQ1.** To what extent do open source developers collaborate on design and modeling artifacts?
- **RQ2.** What is the nature of the model-based collaborations in open source environments?
- **RQ3.** What are the key incentives and barriers for mode-based collaborations in open source systems?

2.1 Subject Systems

The scope of this study includes 62 open source projects selected based on the following criteria. First, we sorted GitHub repositories based on project size and then selected the first 50 most active projects based on GitHub ranking of project activities 11. Second, we selected 11 projects based on the following criteria. Using GitHub advanced search, we identified projects written in C++, Java, JavaScript, Shell, C#, and C. This increases the generality of our results and excludes domain-specific languages that may not represent the general open source practices adequately. We excluded all projects that were not active in the last five years. We excluded projects that did not have at least three active contributors and were not cited in any scientific article on Google Scholar. The citation criterion ensures minimum level of maturity of code and excludes in-progress projects. The resulting set was sorted based on project size, and then we selected the top 11 projects. Thirdly, we added a new system, the Quantum Geographical Information System (QGIS) 5. QGIS was included because it is the premier geo-analysis tool that is developed by both open and closed source developers. It has a global contributor and user base, with a significant interest from private entities that often support professional developers' contributions. The first 50 projects are listed in 1. Table 1 lists the additional 12 projects included in this study.

Table 1. Included open source projects

Num.	Name	Commits	Code Size	Active Contributors
1	Pykep	646	201,430	12
2	Rash	572	148,931	11
3	Epiviz	289	204,528	3
4	Seg3D	2,365	8,574	12
5	BioImageLab	6	15,337	2
6	sead-virtual-archive	408	200,611	8
7	VEGL-Portal	13,33	72,213	5
8	BEACONToolkit	101	156	3
9	mule	61	1,249	2
10	Prov-scaffold	8	2,764	3
11	eo4vistrails	667	18,218	2
12	QGIS	44,029	1.2 m	244

2.2 Survey

We requested short interviews with the survey respondents. When a respondent declined the interview due to time limitation or difficulties in scheduling a suitable

time, we sent out a questionnaire. The questionnaire and the interview discussions were moderated by the following questions:

- **Q1:** What kind of contributions do you make to <project name> (code, test, documentations, other)?
- **Q2:** What is the primary goal or motivation of your contributions (for instance: paid effort, support research you do or someone else is doing, or support commercializing or services)?
- **Q3:** How do you go about understanding the code base to make your contributions? Do you refer to documentations, designs, or do you seek information directly from other developers?
- **Q3:** Is there an overall design, architecture, or model that you refer to? How useful is the design or architecture? Is it up to date? Do you collaborate using models with other contributors?
- **Q4:** In your opinion, what is required to encourage more contributors to the project? What are the key limiting factors?
- **Q5:** Do you consider <tool name> well designed, and the code is of high quality?

3 Results and Analysis

We have examined 62 projects' code, commits, related documentations such as design artifacts and coding standards. We have collected 162 survey responses, conducted six interviews, and collected questionnaire responses from five contributors. Of the interviewed participants, five were paid professionals contributing to the QGIS project. We shared preliminary results and analysis with two participants and conducted two additional follow-up interviews.

3.1 Evidence for design and modeling artifacts

Investigation of the largest 50 open source projects suggests that modeling artifacts are almost non-existent. Based on the number of files, only 0.03% were XML based. Investigation of these resulting files showed that only 0.01% included XMI specific tags. The examination of related documentations, such as development environment setup guidelines, showed that none of these projects has model-based design descriptions. For the other 12 subject projects (shown in Table 1), we found that they contain negligible modeling artifacts. XML files that included XMI specific tags were almost non-existent (less than 0.01%). Related documentations supported the finding that models and design artifacts are not available.

3.2 Evidence for design and modeling practices

Despite the fact that the examination of artifacts does not directly suggest that modeling is practiced, our survey, questionnaire, and interview results suggest a broad set of design and model-based collaborations.

Survey: Participants averaged 10 years of experience, with 50% having more than 5 years of experience, and about 28% having more than 12 years of experience. More than one third of respondents are from the USA. Half of the respondents are from Asia and the rest are from Europe and Africa. 52% of respondents indicated that they either sometimes (42%) or often (10%) engage in design activities on whiteboards. Only 12% indicate that they never use a design tool. Those who participate in design activities reported using a design tool to capture design (78%), transcribe an existing design into a digital format (71%), prototype (60%), brainstorm (45%), and generate some code (72%). 95% of the responses showed interest in using a modeling tool for collaboration. Of those, 60% ranked this capability as very important.

Questionnaire and Interviews: All contributors report code as their primary form of contributions to the open source project. About 27% (3/7) contribute to the test code. Comprehension activities were centered around reading code (95%). Related documentations were not a good source of information for 85% of participants. Interestingly, 36% (4/11) of participants reported engaging in design and model-based collaborations. Those four participants were contacted for follow-up interviewing and we conducted two follow-up interviews. Participants in the interviews were contributors to the QGIS project. Both were professional software engineers compensated for their code contributions. Both participants reported significant design deliberations with other 'key' contributors. For example, one of the participants said: "*we have design documents that I share with my colleagues. We often discuss design decisions in great length.*" Those model-based deliberations are often performed offline using personal and business emails. The primary goal of using design models is to plan work packages and resource assignments.

Code quality is a major concern, but design and modeling approaches do not seem to be the primary approach for improving code quality. This can be seen in this passage: " *.. we need to do much more code reviews, but we do not have the resources for that. But it is in the plans. .. do not see how models can improve code quality. Our models are at a higher level, and we do not translate the models to code.*" Furthermore, there is little deviation from the design specifications and implementations. For instance, we obtain this from one of the participants: "*the code matches the design pretty much.. at least for the core components. The corners [plug-ins developed by open source contributors], it is very different.*"

3.3 Characterization of Model-Based Collaborations

As discussed in Section 4.1, investigation of open source artifacts does not suggest any significant levels of collaborations on models. In this section, we focus on analysis of the survey and questionnaire/interview data.

Survey: Model-based collaborations on whiteboards and during meetings are the most common venues for model-based collaborations. Of the 40% respondents who reported to participate in collaborative modeling regularly, more than 85% perform

these activities on a whiteboard and 54% during meetings. Only 12% share results with close circle of collaborators and none reported publishing results of model-based collaboration along with open source project artifacts.

Questionnaire and Interviews: 36% of participants reported engaging in collaborative design. None of the participants reported using a dedicated design or modeling tool. There was no motivation to use a dedicated modeling or design tool. One said, for example, *"... we do not generate any code or tests from the models."* QGIS is the only project where design deliberations (not the design models themselves) are made publicly available in the form of meeting minutes.

Lack of mechanisms to enforce design specifications in the code seems to be a major factor limiting incentives to share designs. When probed on reasons for not sharing designs, one participants reported *".. I share the designs with three collaborators. They know [the project code] and I can trust they will stick to the design specifications. Why would I share designs if there is no way to enforce it?*" Other factors limiting incentives for sharing designs include relevance to other developers, not being part of the build process, and the casual nature of the available designs, and their change fluidity.

We identified two methods of collaborations, namely asynchronous and synchronous collaborations. In asynchronous collaborations, models are stored in Microsoft Word documents and are shared by emails. Changes are often communicated by chats or emails and are implemented in the model as needed. Multiple copies of the models may exist with different contributors and there is no pressing need to ensure model consistency. In synchronous collaborations, models are stored in the cloud, though often not part of the open source project artifacts. Collaborations were limited to only a few concerned developers. One participant expressed "*.. the design specifications are in the cloud and open for anyone. But .. only a few key developers would [care to / invest time to] contribute to the designs..*" Design deliberations can often be lengthy, and can occur over long periods of time.

3.4 Analysis

Our analysis suggests that model-based collaborations in open source is rather limited. When it is performed, it seems that modeling artifacts are only shared with close collaborators and not shared as part of the open source project artifacts. We term this collaboration style as Champions-only Collaboration. In this style, only a few main contributors (or champions) collaborate on design artifacts. Design artifacts may be made available online, but are typically not available for contributions from the broader set of contributors or users. There is often no documentations or guidance on the available designs. Champions collaborate *offline* on models and other design artifacts. This explains, at least in part, why investigations of open source artifacts often suggest little to no collaborative modeling. Participants in our study indicated lack of incentives to share models beyond the immediate circle of collaborators.

4 Conclusion

We conducted a study to understand the nature of model-based collaboration in open source projects. The study included an investigation and analysis of open source artifacts, a survey, a questionnaire and interviews with open source developers. Our study suggests that model-based collaboration is practiced, but that model-based collaboration artifacts are often not shared as part of the project artifacts. Model-based collaborations are often conducted informally within a small circle of contributors or champions. Designs often do not contribute directly to code and there are little incentives to share design and modeling artifacts beyond the immediate circle of collaborators.

References

1. Badreddin, Omar, Timothy C. Lethbridge, and Maged Elassar. "Modeling practices in open source software." In IFIP International Conference on Open Source Systems, pp. 127-139. Springer, Berlin, Heidelberg, 2013.
2. Franco-Bedoya, Oscar, David Ameller, Dolors Costal, and Xavier Franch. "Open source software ecosystems: A Systematic mapping." *Information and Software Technology*, 91 (2017): 160-185.
3. Ho-Quang, Truong, Regina Hebig, Gregorio Robles, Michel RV Chaudron, and Miguel Angel Fernandez. "Practices and perceptions of UML use in open source projects." In Proceedings of the 39th International Conference on Software Engineering: Software Engineering in Practice Track, pp. 203-212. IEEE Press, 2017.
4. Badreddin, Omar. "Umple: a model-oriented programming language." In Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 2, pp. 337-338. ACM, 2010.
5. QGIS, D. T. (2011). Quantum GIS geographic information system. Open source geospatial Foundation project, 45. Source code available: https://github.com/qgis/QGIS
6. Rahad Khandoker and Omar Badreddin. Professional coding and modeling practices, 2017. Available: https://goo.gl/bQV9Ph
7. Sack, Warren, Françoise Détienne, Nicolas Ducheneaut, Jean-Marie Burkhardt, Dilan Mahendran, and Flore Barcellini. "A methodological framework for socio-cognitive analyses of collaborative design of open source software." *Computer Supported Cooperative Work (CSCW)* 15, no. 2 (2006): 229-250.
8. Badreddin, Omar, Timothy C. Lethbridge, and Maged Elassar. "Modeling practices in open source software." In *IFIP International Conference on Open Source Systems*, pp. 127-139. Springer, Berlin, Heidelberg, 2013.
9. Timothy C. Lethbridge, Andrew Forward, Omar Badreddin. Problems and Opportunities for Model-Centric vs. Code-Centric Development: A Survey of Software Professionals, in the proceedings of C2M:EEMDD 2010.
10. Beller, Moritz, Alberto Bacchelli, Andy Zaidman, and Elmar Juergens. "Modern code reviews in open-source projects: Which problems do they fix?." In *Proceedings of the 11th working conference on mining software repositories*, pp. 202-211. ACM, 2014.
11. GitHub Developer guide. Available: https://developer.github.com/v3/repos/statistics/
12. Nakagawa, Elisa Yumi, Elaine Parros Machado de Sousa, Kiyoshi de Brito Murata, Gabriel de Faria Andery, Leonardo Bitencourt Morelli, and José Carlos Maldonado.

"Software architecture relevance in open source software evolution: a case study." In *Computer Software and Applications, 2008. COMPSAC'08. 32nd Annual IEEE International*, pp. 1234-1239. IEEE, 2008.

13. Gaar, Wolfgang, and Egon Teiniker. "Improving model-based collaboration by social media integration." In *Software Engineering Education and Training (CSEE&T), 2014 IEEE 27th Conference on*, pp. 158-162. IEEE, 2014.
14. Alfayez, Reem, Celia Chen, Pooyan Behnamghader, Kamonphop Srisopha, and Barry Boehm. "An Empirical Study of Technical Debt in Open-Source Software Systems." In Disciplinary Convergence in Systems Engineering Research, pp. 113-125. Springer, Cham, 2018.