LOG6307E – Release Eng. App. Mining Software Repositories

Fall 2023

Paper review of "Studying Pull Request Merges: A Case Study of Shopify's Active Merchant"

Group 01

1949477 – Ming Xiao Yuan

Submitted to: Zohreh Sharafi

October 2, 2023

Summary:

Pull-based development is the concept of forking an existing project, modifying it by either correcting bugs or adding new features, and then pushing their changes to the original repository. The owner of the project can then determine its validity and decide whether to accept it or not. In the scope of this research paper, the authors report on two empirical studies focusing on Pull Requests (PR) for a project called Active Merchant. This is a commercial project developed by Shopify Inc. and contributed to by Spreedly. In this paper, the authors seek to answer three questions: 1) Which merge strategies are used by developers? Do they affect the pull request review time? 2) What factors affect the PR review time and decision? 3) How do developers perform and assess PR reviews? Addressing these problems is important because knowing which merge strategies are popular can inform new or less-experienced developers about best practices. Furthermore, a better understanding of the quality of the submitted code can ensure the quality of the main branch. The main contributions of this paper are: 1) An in-depth study of new contributions in a pull-based software development model. 2) A survey with professional developers that offers insight into their perceptions of the process of assessing new PRs. 3) A publicly shared dataset that includes mined data with manually verified and labelled merged PRs. These contributions are significant because they provide a fundamental understanding of how contributions are made in a pull-based model. By making the dataset public, the paper also contributes to the research community. The results of the quantitative study found that PR size and the number of people involved in the discussion of a PR had a significant impact. Furthermore, the results of the qualitative study show that "developers associate the quality of a PR with the quality of its description, its complexity, and reversibility, while the quality of the review process is linked to feedback quality, test quality, and the discussion among developers." (Kononenko, 2018, p.10).

Positive points:

Points #1: Comparison with other studies

While conducting quantitative and qualitative methods, the authors have listed a few potential factors such as Merge types, Merge time, and Merged decision. In the discussion section, the authors compare their factors with other studies. They have found that for Merge types, developers use squashing more often than other techniques. No past studies have tackled the considerations that developers make when deciding how to integrate a pull request, so it raises an interesting question for future studies. For the Merge time, the authors have different results than Gousios et al. Both studies "showed that developer experience/reputation impacts merge time." (Kononenko, 2018, p.9) The conclusion shows that there is no unified model that works for all Github projects. Finally, for the Merged decision, the authors have the same results as Gousios et al.'s findings. "Both sets of studies showed similar effects of PR size, discussion, and affiliation on the decision to merge a PR." (Kononenko, 2018, p.9) Consequently, it shows the importance of comparing results with prior studies to validate or challenge current results.

Points #2: Realisation of a quantitative and qualitative studies

The authors have made an excellent choice in proving their results with both a quantitative and a qualitative study. For research question #2, the quantitative study shows the readers by employing data mining techniques that "the statistical models revealed both PR review time and merge decision are affected by PR size, discussion, and author experience and affiliation. By conducting a qualitative study, we also see that "PR quality, type of change, and responsiveness of an author are also important factors." (Kononenko, 2018, p.7)

Points #3: Research questions choices

The questions that the authors chose are well-chosen and easily understandable. They cover merge strategies and how they affect the pull request review time, factors that affect the PR review time and decision, and how developers perform and assess PR reviews. In the end, the reader can easily understand the authors' results.

Negative points:

Points #1: Tables are confusing

Tables 3 and 4 are not the easiest to understand. Table 3 shows the models for fitting data. It contains a lot of information about machine learning such as R^2, Tjur's D, VIF analysis, and Stat.significance codes, etc. For a reader with no experience in machine learning and who just wants to learn about factors that affect PR merges, it becomes confusing to understand and interpret the data. Table 4 is too large, which harms readability.

Points #2: Not enough details on specs of the machine learning models

In the methodology section, it describes the data mining algorithms and data pre-processing. It goes over the descriptions very rapidly and does not provide any specific details about the algorithm itself. Additionally, we do not know why the Multiple Linear Regression Model (MLRM) and Logistic Regression Model (LRM) are used, nor the reasons they were chosen to tackle this problem.

Points #3: External validity

In this study, the repository that the authors focus on is Shopify's Active Merchant. Therefore, the data used is focused on one single project developed by experienced full-time software developers. As a result, the authors cannot claim to generalize the results across all projects.

Recommendations:

My first recommendation is to better structure the paper by grouping the quantitative and qualitative studies together. This would make it much easier to find information and improve readability.

My second recommendation is to compile the questions asked to developers into a concise table, rather than scattering them throughout the paper. This would also improve readability and conciseness.

My third recommendation is to include definitions of key concepts, such as “Pull-based development”, so the reader can understand immediately without needing to search for definitions elsewhere.

References

O. Kononenko, T. Rose, O. Baysal, M. Godfrey, D. Theisen, and B. de Water. 2018. Studying pull request merges: A case study of Shopify’s active merchant. In Proceedings of the IEEE/ACM 40th International Conference on Software Engineering: Software Engineering in Practice Track (ICSE-SEIP’18).