CS846-001 Week 11 Reviews

Jun Qing Lim
University of Waterloo
jq3lim@uwaterloo.ca

1 Modern Code Review: A Case Study at Google

1.1 Problem Being Solved

The paper presents a case study on Google's code review practices and processes in the form of exploratory investigation. Code review is a process that is heavily adopted by developers as a method to improve quality of software. However, recent research in code review tend to focus on broad perspective as opposed to narrow perspective (or company-focused), as such this paper aims to provide a comprehensive understanding and insights on code review in a real-world context (particularly, at Google). The authors hope that its findings will provide guidance and insights for software companies looking to improve their code review processes in building better software development practices.

1.2 New Idea Proposed

The authors conducted the study in the form of an exploratory investigation through methods like interviews, surveys, and log data analysis on aspect related to code reviews experience at Google. The findings were then split into the following:

1. On code review motivations
   - the purpose of code reviews is not mainly on code accuracy but also on code readability and maintainability in the long-run
   - code reviews are highly dependent on the relationship between author and reviewers (such that expert will have more code reviews)
2. On code review practices
   - Google's code review flow is tightly coupled with its own internal review tool of CRITIQUE and aligned with the practice of being lightweight and flexible
   - CRITIQUE has reviewer recommendations and code analysis results which increases the speed of code reviews
3. On developer's perception
   - large percentage of developers (97%) are satisfied with the CRITQUE code review process and tool
   - its current code review process is seen as valuable and efficient within Google

The authors finally concluded that Google's code review process is modern and lightweight in comparison with other projects of similar process, which developers find both valuable and good use of their time.

1.3 Positive Points

Firstly, the paper provided a comprehensive analysis on a case study at Google, giving detailed and insightful overview of methods and strategies adopted by Google. Secondly, code review is a practical aspect of software development and the paper's findings are valuable and relevant to the state of software development who are looking to improve their code reivew practices. Thirdly, the authors clearly outline the methodology and objective of the research ranging from interviews to surveys to analysis of code review data, giving readers a good understanding and credibility on how the case study is conducted.

1.4 Negative Points

Firstly, the paper lacks detailed of evidence (or in-depth discussion) and it does not provide strong evidence to support its claim on the effectiveness of the code review process; rather, it is is just a descriptive case study. Secondly, given the renown technology company of Google with large resources to deploy in building an internal code review tool for itself, the findings obtained from this paper may not be useful for other average companies, rather it only serves as an insight from Google's standpoint (i.e., limited generalizability).

1.5 Future Work

We could expand this research by doing a study on how code reviews impact code quality, allowing us to have better understanding on the benefits of a good code review process. Apart from that, we could conduct a study on the differences in the common/top code review processes/practices and its impact, allowing us to have comparison between code reviews in identifying the most effective code review process.

1.6 Rating

4 out of 5. Paper was well-written overall with clear presentation.

1.7 Discussion Points

- What are the benefits of good code review process?
- What are the different current practices of code review in industry?
- Can we automate code reivews? Any tools available at this moment?

2 Characteristics of Useful Code Reviews: An Empirical Study at Microsoft

2.1 Problem Being Solved

The paper presents a study of factors that lead to useful code reviews at Microsoft. Modern code review process has been adopted by many companies as a quality control mechanism. However, many of the comments made by reviewers vary, which affects its usefulness and influences the effectiveness of the code review practices. As such, it is not always clear on what characteristics make a code review useful. The authors hope to identify the traits of useful code reviews to help improve the effectiveness and quality of code review process in software development.

2.2 New Idea Proposed

The authors conducted a three-stage empirical study of code review usefulness at Microsoft as follow:

1. conducted exploratory study to understand meaning of code review comment usefulness
   - interviewed multiple Microsoft developers to survey and rate code review comments
   - more than 60% rated the comments as useful
2. built an automated classifier to distinguish useful and non-useful comments
   - classifier was built by taking in code comments attributes such as keywords, changes, sentiment of comments, etc.
   - model was found to have up to 87% of precision and 94% of recall
3. applied the classifier to 1.5M code review comments
   - used the classifier result to investigate factors on usefulness of code review feedback

The following findings were then obtained:

- reviewers that have experience with an artifact gives more useful comments
- reviewers from different teams give more useful comments than reviewers from the same team
- dentiy of useful comments increases over time (the longer the time of code review, the higher the comment usefulness)
- code reviews with larger files have less useful comments
- source codes have the highest comment usefulness than other file types

2.3 Positive Points

Firstly, the authors outlined a clear methodology in conducting the empirical survey, allowing the methods to be replicated / referenced to other studies. Secondly, I liked how the paper used statistics to convince the readers on the findings, making the results to be more credible. Thirdly, the findings are useful, practical and relevant to the industry, as knowing the characteristics gives better awareness to developers on improving their code review practices in software development field.

2.4 Negative Points

Firstly, as the research was conducted based on interviews (data from developers), as such the findings may be subjected to personal biases / experiences. Secondly, the research scope included data from limited teams from Microsoft, as such it may not be fully representable for all teams and the code usefulness across other genre / domains of teams may vary.

2.5 Future Work

We could extend this project by doing an extended study of impacts of code review characteristics in software development, allowing us to better understand the role of useful code reviews in the field. Also, we could conduct code reviews study in different contexts such as open-source projects or startups to better understand its practices in different environment.

2.6 Rating

4 out of 5. The paper gave great insights that would impact the awareness of software developers in providing a useful code review comment to improve the productivity and efficiency of software development process.

2.7 Discussion Points

- What is the general process for a code review?
- How does code review process and practices impact software developers?
- What constitute a good code review?
- Any possible biases from this paper?