Impact of Code reviews on Software Code Quality

Ganna Anil gaam17@student.bth.se 960710-7514

Introduction:

This assignment mainly consist of summaries , main finding , study on how authors define code quality and reflections from studying papers of 5 different authors related to code smells/technical debts.

Summaries :

1) This paper mainly focused on finding how code smells effect the maintainability of the code which are indentified as main code smells by the programmers. There many studies that examined the relationship between code smells and maintainability . in this paper they tried to answer on how code smells can be used estimate the maintainability through a case study. They conducted the case study by hiring set of developers to maintain the code for 14 days and also conducted daily interviews to find perception of programmers on the code smells. After the case study the author compared results from his case study with finding of Anda (who made on study on code smells that effect the maintainability

2) The paper mainly focused on prioritizing the technical work to be done. This paper comes up with an idea of using cost/benefit analysis to prioritize the technical debts to reduce the work done. In this paper they tried to come up with an approach how technical debts can be prioritized by giving set of debt issues.

3) This paper focused on finding the relationship between the code smells and software faults. And also they tried to prioritize which code smell should be refactored based on their effect on the software. In this paper they consider six of the Fowler et al.'s code smells that are Duplicated Code, Data Clumps, Switch Statements, Speculative Generality, Message Chains, and Middle Man to prioritize. They considered duplicate code contains more number of faults so they have given first priority to duplicate code and prioritized remaining 5 code smells based on their effect.

4) This paper is summary of third international workshop held on managing technical debts. In this workshop they mainly focused on eliciting and visualizing debt and payback strategy. In this workshop they focused on two aspects 1) understand to process and indicators of technical debt and 2) examining different payback strategy to analyze how technical debts can be handled.

5) This paper tried to find the relationship between the software quality , code review and code review participation. They conducted case study on three different products in which they found code review and no of times reviewers interference in code has impact on code quality.

CODE QUALITY:

1) In this paper the author described that code smell is one of the subprime choice that decreases the quality of the code. Some of the characteristics like code size also effects the maintainability of the code.

2) Author addressed too much increase in the technical debt effect the software development which is due to poor increasing in maintainability of code and also code quality decreases of the prioritizing main concentrate on decreasing the cost.

3) Author addressed that evolvement of software increases the requirement which increases the complexity of the source code. As the complexity increases quality of code increases with increase in number of code smells.

4) Author addressed that developers and managers frequently disagree on few aspects which include code and design quality which are invisible to customer and management.

5) Author addressed that from the study of Nonetheless, Bacchelli and Bird , code review is one of the best way to improve quality before or after integration of software.

MAIN FINDINGS:

1) The main findings of this papers are code smells provide good observation of maintainability and also they found that some of the factors effecting maintainability require alternative methods to evaluate as they are not reflected by the code smells. They also found new factors that effects the maintainability from interviews which are not reported by previous researchers.

2) The main finding of this paper are how technical debt can be prioritized based on the likelihood of their change and defect. It also found that summing up technical debts can also effect the quality of the code.

3) The main findings of this paper are as considered in this begging duplicate code should be given priority as it is collaborate with more number of faults. And out of remaining code smells message chain should be given more priority as it has more number of faulty out of all the 5 code smells.

4) In the workshop they found that the tools used does not detect some of the debts and also technical debts should not be treated separated from new faults. Taking wrong decisions is not only reason for technical debts.

5) The main finding of this paper are code review metrics can only contribute to two out of four defects and only 2 of three reviewers contribute for the code. And also found relation between number of code and number of post delivery defects in a component.

LEARNINGS:

1) This paper helped me to learn how different factors effecting the maintainability are covered by different code smells. And also how many different code smells are automatically detected and alternative methods to detect different code smells.
2) This paper helped me in learning how prioritizing the technical debts can help in reducing the work and also helped me to learn how technical debts impacts the code quality.
3) Learned how to identify different type of code smells in a source code using different tools and also how to prioritize the code smells using binary coding schema.
4) Learned that evolution of new technologies, commercial success also cause technical debts in software development.
5) Learned how code review and code reviewers effect the code quality and also learned that components having less number of review have high post release defects.

REFERENCES:

1) Yamashita, Aiko, and Leon Moonen. "Do code smells reflect important maintainability aspects?." *2012 28th IEEE international conference on software maintenance (ICSM)*. IEEE, 2012.
2) Zazworka, Nico, Carolyn Seaman, and Forrest Shull. "Prioritizing design debt investment opportunities." *Proceedings of the 2nd Workshop on Managing Technical Debt*. ACM, 2011.
3) Zhang, Min, et al. "Prioritising refactoring using code bad smells." *Software Testing, Verification and Validation Workshops (ICSTW), 2011 IEEE Fourth International Conference on*. IEEE, 2011.
4) Kruchten, Philippe, et al. "Technical debt in software development: from metaphor to theory report on the third international workshop on managing technical debt." *ACM SIGSOFT Software Engineering Notes* 37.5 (2012): 36-38.
5) McIntosh, Shane, et al. "The impact of code review coverage and code review participation on software quality: A case study of the qt, vtk, and itk projects." *Proceedings of the 11th Working Conference on Mining Software Repositories*. ACM, 2014.