MGTE 31222 - Research Methods (21/22)

Paper Review

"An Empirical Evaluation of the Impact of Refactoring on Internal and External Measures of Code Quality"

Group 04

IM/2019/010 - Hiruni Keenavinna

IM/2019/041 - Hasali Edirisinghe

IM/2019/092 - Amaya Senarathne

IM/2019/096 - Harini Jayasundara

The paper titled "An Empirical Evaluation of the Impact of Refactoring on Internal and External Measures of Code Quality" by S.H. Kannangara and W.M.J.I. Wijayanake raises important issues regarding the impact of refactoring on code quality and attempts to provide quantitative evidence to support their claims. However, there are several concerns with the methodology and the conclusions drawn from the study.

The main issues raised in the paper include the lack of quantitative evidence supporting the improvement in software quality due to refactoring, the contradictory findings in previous studies, the need for evaluating the impact of different refactoring techniques, and the reliance on internal and external quality measures. The authors aim to address these issues by conducting an empirical study that evaluates the effect of refactoring on code quality using both internal and external measures. The conclusion of the research, based on the experimental results, indicates that there was no significant improvement in code quality observed after applying refactoring techniques, particularly when considering the external quality factors.

The research methodology employed in the study is a quantitative approach with an experimental research design. The authors formulated four research hypotheses related to different aspects of code quality and conducted experiments to test these hypotheses. The study involved applying ten selected refactoring techniques to a small-scale project and measuring both internal metrics (e.g., cyclomatic complexity, lines of code) and external metrics (e.g., analysability, changeability).

However, there are several problems with the methodology that limit the validity and generalizability of the results. Firstly, the absence of a control group makes it challenging to attribute any changes in code quality solely to refactoring. The study only focuses on a small-scale project with bad smells, which may not adequately represent real-world software systems. Additionally, the limited selection of refactoring techniques and the reliance on specific quality models and measures may not capture the full range of refactoring impacts and code quality aspects. The lack of contextual information further hampers the interpretation of the results.

The experimental results comparing pre-refactoring and post-refactoring code metrics, as well as comparisons with earlier studies, are presented by the authors as evidence to back up their claims.

Although evidence is provided, it is important to note that the conclusions drawn from this evidence are based on specific metrics and may not necessarily apply to all aspects of code quality. Moreover, there are discrepancies between the findings of this study and previous studies, indicating that the impact of refactoring on code quality may vary depending on the specific context and metrics used.

In terms of agreement, it is reasonable to agree with the authors' use of internal and external measures to assess the impact of refactoring on code quality. The consideration of raw metric values for a more reliable conclusion is also a valid approach. However, there is disagreement with the authors' conclusion that there is no quality improvement in source code after refactoring, as it is based on specific metrics and may not generalize to all types of refactoring or other metrics of code quality.

To address the limitations and enhance the understanding of the impact of refactoring on code quality, several valuable alternatives were proposed. Firstly, conducting a study with a larger sample size would strengthen the reliability of the results, mitigating concerns about the limited number of participants. Additionally, involving expert developers in the evaluation process could provide insights based on their extensive experience and knowledge of system design. Assessing commonly used refactoring techniques in the industry through surveys would offer practical insights applicable to real-world software development scenarios. Moreover, exploring additional code quality metrics, such as code readability, maintainability, and error rates, would provide a more comprehensive assessment of the effects of refactoring. By considering these alternatives, future research endeavors can overcome the limitations of the present study and yield more robust conclusions regarding the impact of refactoring on code quality.

In conclusion, while the paper raises important issues and presents a research methodology for evaluating the impact of refactoring on code quality, there are significant limitations and concerns with the methodology and conclusions. Further research and exploration of alternative approaches are necessary to obtain a more comprehensive understanding of the effects of refactoring on code quality.