Grant Proposal

Jincheng He
Department of Computer Science
University of Southern California
jinchenh@usc.edu

I. Introduction

Developing software with the source code open to the public is very common; however, similar to its closed counterpart, open-source has quality problems, which cause functional failures, such as unsatisfying user experience, and non-functional, such as long responding time. To improve the quality of open source software, we investigate how it is impacted by commits of different purposes. By identifying these impacts, we will establish a new set of guidelines for commiting changes, thus improving the quality.

II. Previous work

Previous researchers have revealed when, where, how and what the developers contribute to projects and how these aspects impact software quality. However, there has been little work on how different categories of commits impact software quality.

III. Our Plan

A. Stage One

Few researchers have studied the correlation between the change type and the code, or how change type impacts quality. In this stage, we start by refining the existing categorization of commit changes in open source software repositories. We evaluate the quality of those changes by obtaining quality metrics from static analysis tools. To assess the correlation between the quality and the categories, we will train a machine learning model, in addition to applying standard mathematical correlation analyses.

B. Stage Two

In this stage, although we have categorized the commit changes, further work distinguishing between different categories is required. This is because high-level categories overlap. In this stage, we will remove the ambiguity of the categories by analyzing the code changes within the commits rather than the commit messages and manual categorizing. Once this is done, we will investigate the correlation between the categories and changes in code to reveal whether they correlate and how those changes impact software quality.

C. Stage Three

In the final stage, we will construct guidelines which will help developers develop software. In addition, we will create an index which explains how different code patterns impact the quality. We will conclude this research by completing and validating these two aspects.

IV. Feasibility

This project is feasible because the requisite data, tools and techniques are readily available:

- Data: Open-source software and Git provide sufficient meta-data from Google, Netflix and Apache projects.
- Tool: PMD, SonarQube, FindBugs and CAST provide various quality metrics.
- Techniques: Machine learning and natural language methods.

With all above, we believe this plan will succeed in three to five years. The midterm milestone is the reasonably high prediction accuracy from the machine learning model. The final milestone is the completion of the new systematic coding standard and development guidelines.

V. Benefits

We will be able to provide guidelines on how open-source software developers, when contributing to projects, can improve quality. In addition, the results of the second stage will allow us to provide more reliable coding standards and will improve overall code quality. Improved quality will help to reduce cost and improve software service quality.

VI. Intellectual Advancement

The first goal of this research is to make concrete improvements in the quality of open-software by providing guidelines for developers and, thus, to improve code quality. We believe this will also change the way people think, code and develop software.