International Journal of Open Source Software and Processes
Volume 11 • Issue 1 • January-March 2020

Ripple Effect Identification in Software Applications

Anushree Agrawal, Indira Gandhi Delhi Technical University for Women, Delhi, India
https://orcid.org/0000-0002-2811-6780

R.K. Singh, Indira Gandhi Delhi Technical University for Women, Delhi, India
https://orcid.org/0000-0001-8729-2293

ABSTRACT

Changes are made frequently in software to incorporate new requirements. The changes made to one class are not limited to that particular class, but they also affect other entities. Early identification of these change prone entities is very essential for minimizing future faults in the software applications. Thus, it is very important to develop quality models for identifying the ripple effect of changed classes to effectively utilize the limited resources during the software development lifecycle. Association rule mining is a popular approach suggested in literature, but a major limitation of this approach is its inability to generate recommendations in case of new addition of classes. This article suggests the development of prediction model using learning techniques to overcome this limitation. The authors evaluate the performance of thirteen statistical, ML, and search-based techniques using eight open source software applications in this work. The findings of this study are promising and support the application of SBT and ML techniques for ripple effect identification.

KEYWORDS

Co-Change Prediction, Friedman Test, Machine Learning, OO Metrics, Search-Based Techniques

INTRODUCTION

Software systems undergo numerous changes throughout their life span. Rectification and enhancements are two main factors responsible for these changes made to software applications. Software maintenance is a very important phase in the development life cycle and it has been observed that it employs the lion's share of the total allocated expense. Researchers have found that ripple effect is capable of picturing the extent of change effect on the rest of the software application (Arvanitou, Ampatzoglou, Chatzigeorgiou, & Avgeriou, 2015). Black (2006) have studied the role of ripple effect in software evolution and they found it very beneficial in the software maintenance phase. "It can highlight modules with high ripple effect as possible problem modules which may be especially useful in preventive maintenance. It can show the impact in terms of increased ripple effect during perfective and adaptive maintenance where the functionality of a program is being modified or its environment has changed," according to Black (2006).

Ripple effect identification aims to classify those classes that are modified as aftereffects of changes made to a software application. These change prone classes demand allocation of additional resources like labor, budget and time for their thorough testing and meticulous tracking. Good

DOI: 10.4018/IJOSSP.2020010103

planning and allocation help in minimizing errors leads to good quality software. Thus, it helps in better optimization and proficient exploitation of prevailing resources.

In this paper, we propose development of quality models by establishing the correlation amongst object-oriented (OO) metrics and ripple effect of pair of classes. The proposed model is developed using learning techniques. Different learning techniques (statistical, machine learning [ML] or Search Based Techniques [SBT]) are available to our disposal. The selection of effective learning technique is a major challenge. Agrawal and Singh (n.d.) studied the correlation amongst OO software metrics and ripple effect recently. Prediction model was derived using statistical technique in this work. Performing a thorough search of relevant literature, we found that the potential of other learning techniques such as ML and SBT has not been explored yet for ripple effect identification studies.

ML applies instructed procedures that accept and examine entered data to forecast output values within limits. When new data is provided as input to these procedures, they learn and adjust their functions for improved results and thus become intelligent with the passage of time. SBT are one of the most emerging practices in the recent years. They perform iterative search across the complete solution universe and identify the optimum or near optimum solution. It starts with set of candidate solutions and calculate the fitness for each candidate. Mutation, crossover, reproduction or other selection techniques are then applied to select the best candidate for future generations (Eiben & Smith, 2003). Until the desired quality of results is obtained, the algorithm goes on generating future generations. Several studies in literature support the usefulness of SBT to solve software engineering problems that have certain inconsistencies, no perfect answer or no pinpoint rules for effective solutions (Harman & Jones, 2001). The ripple effect identification for change prone classes reveals similar behavior. There are no precise rules and at times insufficient data when identifying ripple effect of changed classes in software applications.

We study the efficiency of the statistical technique (Logistic regression), five ML and seven Search Based Techniques for ripple effect identification and compare their results using non-parametric test. We evaluate the ten-fold validation results for validating the proposed prediction models developed with different learning techniques. The following research questions are explored in this paper:

RQ1: Are the OO metrics of classes associated with the ripple effect?
RQ2: Can we use SBT and ML techniques for ripple effect identification problems?
RQ3: How is the relative performance of different techniques for ripple effect identification of changed classes?

The data for our study is collected from eight open source software applications developed in java to investigate the answers to the above research questions. The prediction model is built using each technique and then we compare their performance with the help of ROC measures. Friedman statistical non-parametric test is applied to evaluate the relative performance of proposed prediction models. The study reveals that the performance of most of the ML and SBT based models is better than the ones provided by statistical technique. Moreover, the results obtained from SBT and ML techniques are comparable to each other.

The organization of the rest of the paper is as follows. The relevant related literature is discussed in section 2. The research methodology is presented in section 3. Section 4 states the empirical data collection; Section 5 specifies the results of our study. Section 6 presents the conclusions of our study and the future study directions.

RELATED WORK

The changeability sub-characteristic indicates the ease of modifying software. It relates metrics for quantifying the software attributes that conclude about the effort required for alteration, environmental change or defect elimination (ISO/IEC 9126-1, 2001). Change coupling identification is an important

Related Content