A Developer Centered Bug Prediction Model

ABSTRACT

Several techniques have been proposed to accurately predict software defects. These techniques generally exploit characteristics of the code artefacts (e.g., size, complexity, etc.) and/or of the process adopted during their development and maintenance (e.g., the number of developers working on a component) to spot out components likely containing bugs. While these bug prediction models achieve good levels of accuracy, they mostly ignore the major role played by human-related factors in the introduction of bugs. Previous studies have demonstrated that focused developers are less prone to introduce defects than non-focused developers. According to this observation, software components changed by focused developers should also be less error prone than components changed by less focused developers. We capture this observation by measuring the scattering of changes performed by developers working on a component and use this information to build a bug prediction model. Such a model has been evaluated on 26 systems and compared with four competitive techniques. The achieved results show the superiority of our model, and its high complementarily with respect to predictors commonly used in the literature. Based on this result, we also show the results of a “hybrid” prediction model combining our predictors with the existing ones.

**EXISTING SYSTEM**

* The Chidamber and Kemerer (CK) metrics [36] have been widely used in the context of bug prediction. Basili et al. [1] investigated the usefulness of the CK suite for predicting the probability of detecting faulty classes. They showed that five of the experimented metrics are actually useful in characterizing the bug-proneness of classes.
* The same set of metrics has been successfully exploited in the context of bug prediction by El Emam et al. [26] and Subramanyam et al. [27]. Both works reported the ability of the CK metrics in predicting buggy code components, regardless of the size of the system under analysis.
* Ohlsson et al. [3] focused the attention on the use of design metrics to identify bug-prone modules. They performed a study on an Ericsson industrial system showing that at least four different design metrics can be used with equivalent results. The metrics performance are not statistically worse than those achieved using a model based on the project size.
* Zhou et al. [29] confirmed their results showing that size-based models seem to perform as well as those based on CK metrics except than the Weighted Method per Class on some releases of the Eclipse system. Thus, although Bell et al. [35] showed that more complex metric-based models have more predictive power with respect to size-based models, the latter seem to be generally useful for bug prediction.

**Disadvantages**

* + There is no Product and process metrics Technique in this system.
  + There is no technique called Structural scattering to find bugs effectively.

**PROPOSED SYSTEM**

* The Proposed system is extended the empirical evaluation of our bug prediction model by considering a set of 26 systems.
* Compare our model with two additional competitive approaches, i.e., a   
   prediction model based on the focus metrics proposed by Posnett et al. [22] and a prediction model based on structural code metrics [24], that together with the previously considered models, i.e., the BCCM proposed by Hassan [8] and the one proposed by Ostrand et al. [9] [10], lead to a total of four different baselines considered in our study.
* Devise and discuss the results of a hybrid bug prediction model, based on the   
  best combination of predictors exploited by the five prediction models   
  experimented in the paper.
* Provide a comprehensive replication package [25] including all the raw data   
   and working data sets of our studies.

**Advantages**

* This system implements Research Questions and Baseline Selection which is effective in fining bigs.
* The system has a technique to Detect bugs of Mining Software Repositories.

**SYSTEM REQUIREMENTS**

➢ **H/W System Configuration:-**

➢ Processor - Pentium –IV

➢ RAM - 4 GB (min)

➢ Hard Disk - 20 GB

➢ Key Board - Standard Windows Keyboard

➢ Mouse - Two or Three Button Mouse

➢ Monitor - SVGA

**Software Requirements:**

* Operating System - Windows XP
* Coding Language - Java/J2EE(JSP,Servlet)
* Front End - J2EE
* Back End - MySQL