

# Chapter 7

## Important Issues in Software Fault Prediction: A Road Map

**Golnoush Abaei**

*University Technology Malaysia, Malaysia*

**Ali Selamat**

*University Technology Malaysia, Malaysia*

### ABSTRACT

*Quality assurance tasks such as testing, verification and validation, fault tolerance, and fault prediction play a major role in software engineering activities. Fault prediction approaches are used when a software company needs to deliver a finished product while it has limited time and budget for testing it. In such cases, identifying and testing parts of the system that are more defect prone is reasonable. In fact, prediction models are mainly used for improving software quality and exploiting available resources. Software fault prediction is studied in this chapter based on different criteria that matters in this research field. Usually, there are certain issues that need to be taken care of such as different machine-learning techniques, artificial intelligence classifiers, variety of software metrics, distinctive performance evaluation metrics, and some statistical analysis. In this chapter, the authors present a roadmap for those researchers who are interested in working in this area. They illustrate problems along with objectives related to each mentioned criterion, which could assist researchers to build the finest software fault prediction model.*

### 1. INTRODUCTION

As today's software grow rapidly in size and complexity, the prediction of software reliability plays a crucial role in a software development process (Catal, 2011). Software fault<sup>1</sup> is an error situation of the software system. The explicit and potential violation of security policies at runtime causes these faults

DOI: 10.4018/978-1-5225-3923-0.ch007

because of wrong specification and inappropriate development of configuration (Dowd, MC Donald, & Schuh, 2007). Early detection of fault-prone software components could enable verification experts to concentrate their time and resources on the problematic areas of the system under development. In fact, when limited budget and time do not allow for complete testing of an entire system, software managers can use predictors for software quality to focus on testing parts of the system that seems to be defect prone. So the defect prone trouble spots can then be examined in more detail by model checking and intensive testing. In fault prediction approaches, previous reported faulty data with the help of distinct metrics could identify the fault-prone modules. In addition, important information about location, number of faults and distribution of defects are extracted to improve test efficiency and software quality of the next version of the program. In other word, fault prone modules can be identified prior to system test by using these models. Application of defect predictors in software developments helps the managers allocate the resources such as time and effort more efficiently. It would also be cost effectively to test some certain section of the code (Calikli, Tosun, Bener, & Celik, 2009).

Area of software fault prediction still poses many challenges and unfortunately, none of the techniques proposed within the last decade has achieved widespread applicability in the software industry due to several reasons including the lack of software tools to automate this prediction process, the unwillingness to collect the fault data, and other practical problems (Catal, Sevim, & Diri, 2009).

According to (Catal & Diri, 2009b), analyzing and predicting defects are needed for three main purposes, firstly, for assessing project progress and plan defect detection activities for the project manager. Secondly, for evaluating product quality in order to assess the finished product, and thirdly, for improving capability and assessing process performance for process management.

Any fault prediction model could be built based on variety of learning methods, distinctive software metrics, and different performance evaluation metrics; each of them has its own challenges. We should mention that the type of applications that software metrics extracted from them could also be varied. In this chapter, we reviewed all related parameters to build a finest prediction model. Although literature review paper (Catal, 2011) and many systematic literature review papers (Arisholm, Briand, & Johannessen, 2010; Catal & Diri, 2009b; T. Hall, Beecham, Bowes, & Counsell, 2011; Radjenović, Heričko, Torkar, & Živković, 2013) are available, but to our knowledge none of them has classified the fault prediction issues and challenges as we have done here. Figure 1 shows the overall structure of building any software fault prediction model. All problems and objectives related to each part of this diagram are discussed and reviewed throughout this chapter.

Often, fault prediction approaches are most suitable in less critical domains such as software packages and telecommunication companies. Anyhow, in limited time and budget and specially for testing different versions of the software that have been completely tested before, using software prediction model would be a good choice. One of the practical examples could be found in Chapter 24 entitled “Building Defect Prediction Models in Practice”. Key questions and consequences arising when establishing defect prediction in a large software development project are answered in the mentioned chapter.

This chapter is organized as follows: section 2 presents background of the problem. Section 3 describes different fault prediction issues. Related works are presented in section 4. Section 5 presents fault prediction challenges and finally section 6 is conclusion.

27 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the publisher's webpage:

[www.igi-global.com/chapter/important-issues-in-software-fault-prediction/192877](http://www.igi-global.com/chapter/important-issues-in-software-fault-prediction/192877)

## Related Content

---

### IT-Driven Business Model Innovation: Sources and Ripple Effects

Sune Müller and Mads Hundahl (2020). *Disruptive Technology: Concepts, Methodologies, Tools, and Applications* (pp. 165-190).

[www.irma-international.org/chapter/it-driven-business-model-innovation/231187](http://www.irma-international.org/chapter/it-driven-business-model-innovation/231187)

### Cyber Security Risks in Robotics

Ishaani Priyadarshini (2018). *Cyber Security and Threats: Concepts, Methodologies, Tools, and Applications* (pp. 1235-1250).

[www.irma-international.org/chapter/cyber-security-risks-in-robotics/203558](http://www.irma-international.org/chapter/cyber-security-risks-in-robotics/203558)

### ECSE: A Pseudo-SDLC Game for Software Engineering Class

Sakgasit Ramingwong and Lachana Ramingwong (2018). *Computer Systems and Software Engineering: Concepts, Methodologies, Tools, and Applications* (pp. 191-205).

[www.irma-international.org/chapter/ecse/192878](http://www.irma-international.org/chapter/ecse/192878)

### Relationship Between Knowledge Management and Innovation

Andrea Bencsik and Bálint Filep (2020). *Disruptive Technology: Concepts, Methodologies, Tools, and Applications* (pp. 531-554).

[www.irma-international.org/chapter/relationship-between-knowledge-management-and-innovation/231204](http://www.irma-international.org/chapter/relationship-between-knowledge-management-and-innovation/231204)

### Visualization: Future Technology and Practices for Computational Science and Engineering

Joanna Leng, Theresa-Marie Rhyne and Wes Sharrock (2012). *Handbook of Research on Computational Science and Engineering: Theory and Practice* (pp. 381-413).

[www.irma-international.org/chapter/visualization-future-technology-practices-computational/60368](http://www.irma-international.org/chapter/visualization-future-technology-practices-computational/60368)