A FUTURE APPROACH FOR MODEL-BASED TESTING: ISSUES AND GUIDELINES

Ravinder Veer Hooda
Assistant Professor (CSE Deptt.)
Emax Institute of Engg. & Technology, Ambala (Haryana), India

Abstract: While choosing the behaviour model to be used in a MBT approach, we have to analyse its limitations, and what it can or cannot represent. This information may limit the use of this MBT approach.

Keywords – Testing, MBT approach

INTRODUCTION

This research paper analyses the contexts in which MBT has not been frequently applied. Subsequently, a strategy to define one approach to be applied in one of these contexts based on its specific characteristics. To perform that, we need to know what characteristics are important in a MBT approach, and how to relate the MBT characteristics with the specific characteristics of the software domain and the testing level that the approach to be developed will be applied.

FUTURE APPROACH FOR RESEARCH IN MODEL-BASED TESTING

Three open questions after going through research till date are suggested for future work. These questions are:

1. What are the contexts (testing levels, software domains, etc.) in which Model based testing strategies have not been applied? And why?
2. How are non-functional requirements descriptions used for test cases generation?
3. What are the defects types not detected by test cases generated from specific model?

WHAT CONTEXTS MODEL-BASED TESTING STRATEGIES NOT BEEN APPLIED?

Traditional testing techniques (not based on models) are published on the technical literature for several software domains and testing levels. However, MBT approaches have not been frequently applied for some contexts (Figure 1). MBT approaches have not been applied and analysed, what are the reasons/characteristics/limitations that prevent the use of MBT approaches.

Some issues that may be used as starting point for the research that evaluates MBT for different contexts are presented below:

- How to create a single MBT approach for use at different testing levels during the entire software development process?
- How to reduce the test case set without decreasing the testing coverage (i.e. eliminating redundant test cases)?
- What are the main characteristics of software domains that MBT approaches have not been frequently applied to? These domains include Product Line Software, COTS, Web Service, and Web application). How to test applications in these software domains using a MBT approach?
- What models may be used to describe the specific characteristics for the above domains? What are testing coverage criteria to be applied for applications in each of the above domains?
- Why tool usage is for MBT based Integration Testing low? And how increase MBT tools for Integration Testing?

These issues have similar purposes. Therefore, the guidelines towards a solution for them could be generalized in some steps:
• Observe the main characteristics for each context where MBT has not been frequently applied yet.
• Propose or identify an interesting method or approach to test an application in this context.
• Define how to use an approach based on model to test an application in this context.
• Execution of Experimental Studies to evaluate the effectiveness of this solution for different characteristics, like effort, time, coverage, cost.

HOW ARE NON-FUNCTIONAL REQUIREMENTS DESCRIPTIONS USED FOR TEST CASES GENERATION?

Software Requirements Specification document consists of Functional and Non-functional requirements (IEEE 830, 1998). Functional requirements describe the software components or modules to be developed during the software design and coding phases and to be used to test case generation for system or integration level testing. Non-functional requirements are essentials to define architectural aspects and constraints to be addressed during the software development. Both need to be modelled and tested.

However, the majority of MBT approaches use only functional requirements during test case generation process. Non-functional requirement descriptions have not been frequently used for test case generation.

Figure 2. Usability of Functional and Non-functional Requirements in the software development and MBT process.

From these questions, an approach for test cases generation for different type of non-functional requirements for different software domain (according with some standard, like ISO or IEEE) can be developed. Some issues based on to use non-functional requirements descriptions for MBT are presented below:

• How to classify non-functional requirement for a specific software domain?
• How to model quality attributes for application in a specific software domain?
• Which model to use for non-functional requirement behaviour modeling?

• How to measure the testing coverage about non-functional requirement?
• Which type of non-functional requirement can be tested by MBT approaches?

These issues are integrated with the purpose of make able the use of non-functional requirements for a MBT solution. Therefore, the guidelines towards this solution could be described in some steps:

• Define the context that non-functional requirements descriptions will be used for MBT.
• How to classify non-functional requirements in the selected software domain? For example:
• How to test non-functional requirement for the selected software domain? For example:
• Which model to use for non-functional requirements modeling? For example:
• Propose a method to test types of non-functional requirements for a software domain.
• Execution of Experimental Studies to evaluate the effectiveness of this solution based on different characteristics, like effort, time, coverage, and cost.

WHAT ARE THE DEFECTS TYPES NOT DETECTED BY THE TEST CASES GENERATED FROM A SPECIFIC BEHAVIOUR MODEL?

Tests can show failures in the software, but not the absence of failures. The defects could remain in the software after the tests. Software development is a human-task and consequently errors may be made during this process. Testing try to show fails in a product to support the elimination of the defects that start these fails. However, to reach 100% of testing coverage is a hard and impracticable task; since we had to test the universe of all possible inputs.

Figure 3. Behavior of testing coverage criteria applied to try to discover defects in software.

Testing coverage criteria aim to define strategies to assure an acceptable quality level for software without to reach 100% of coverage. However, eventually, some types of defects don’t get to be discovered by a testing approach. This problem is more frequently in approaches based on models, since that they are dependent on the model used to describe
Discover types of defects not detected by MBT approaches and to propose improvements on the testing process to avoid these defect types in future projects. This solution may be using a MBT solution or not. The goal is to increase testing coverage, and also eventually decrease effort, time, and cost to use it. Apply a causal-analysis and resolution approach to establish the relation between defects and their cause. New strategies may be defined to avoid this type of defects in future projects, using a MBT approach or not. The issues are:

- What were the types of defects detected using a MBT approach?
- What were the types of defects not detected using a MBT approach, but detected after the deployment?
- How to extract metrics from the software development process about the MBT approach used during it?
- How is the defects distribution per phase or per type of defects?
- What are the origins of these defects? (Limitations of the model? Testing criteria? Tool? Testing process? Modelling error?)
- How to avoid these types of defects in future projects using a MBT approach or other strategy to support the testing?

These issues involve a strategy to support the evaluation and improvement of MBT approaches. Other purpose is to support the technology transfer from academic environment to industry.

The guidelines towards this solution could be described in some steps:

- Define metrics to be extracted from a software development projects about MBT approaches.
- Collect information about MBT approaches in software projects to be analyzed.
- Develop a causal-analysis and resolution approach to know the origin of these defects in different approaches. For example:
- Analyze the types of defect usually not identified by MBT approaches and propose a Characterization of Model-based Software Testing Approaches Systematic Review. This document has been developed to be a joint SCR/COPPE Technical Report at PESC/COPPE/UFRJ - Brazil 75 solution (based on model or not) to avoid them.
- Execution of Experimental Studies to evaluate the improvement of the new developed strategy.

CONCLUSIONS

This paper discussed about the improvement of a MBT approach or just a testing strategy using information extracted from software projects. To apply a causal analysis and resolution approach to establish the relation between defects and their cause in an organization and to evaluate the application of a MBT approach in a real project. From this information, a new strategy to improve the testing process and to avoid these types of defects not found by MBT approaches in future projects may be developed.

REFERENCES