Regression Testing Using UML Models: A Survey

Jayanti Khatri
Department of Computer Science, INDIA

ABSTRACT
Regression testing is type of retesting of a system or component to verify that modifications have not caused unintended effects and that system and component still valid with its specified requirements. As it is often expensive and time consuming process, many techniques have been proposed for regression testing that suggest testers how to build and select test suites from existing test suite in less time with minimum cost. In this paper we discuss the advantages and limitations of using Unified Modeling Language (UML) models for regression testing and analyze that UML diagrams are best to use for generating and selecting test cases for regression testing. Our objective is to survey the existing UML based regression testing techniques.

Keywords: Regression Testing, test selection, UML models

I. INTRODUCTION
Software systems undergo various modifications throughout their lifetime which are essential to accommodate user requirements and new technologies. But these changes can adversely affect the quality and reliability of the software. Thus software needs to be revalidated after changes and software test engineers perform regression testing to revalidate the software. Many techniques have been proposed for regression testing like Code based techniques [1, 2, 3], Model based techniques (UML based techniques) [4, 5, 6, 7, 8, 9, 10, 11] etc. Unified Modeling Language (UML) based techniques have proved to be one of the best because of their advantages [12] over other techniques. Regression testing is defined [13] as the process of retesting the modified parts of the software and ensuring that no new errors have been introduced into previously tested code. Let P be a program, let P’ be modified version of P and let T be a test suite for P. Regression testing consists of reusing T on P’, and determining where new test cases are needed to effectively test code or functionality added to or changed in producing P’. Some regression testing techniques [13] are: Retest all, Regression Test Selection, Test Case Prioritization, and Hybrid Approach.

➢ Retest All: It is one of the conventional methods for regression testing in which all the tests in the existing test suite are rerun. So the retest all technique [14] is very expensive.
➢ Regression Test Selection: In this technique selected part of test suite is to rerun. It divides the existing test suite into- Reusable test cases, Retestable test cases and obsolete test cases. Regression test selection [4] can be based on source code control flow and data flow analysis.
➢ Test Case Prioritization: It prioritize the test cases so as to increase a test suite's rate of fault detection that is how quickly a test suite detects faults in the modified program to increase reliability. There are two types of prioritization: (1) General Prioritization [15] which attempts to select an order of the test case that will be effective on average subsequent versions of software, (2) Version Specific prioritization which is concerned with particular version of the software.
➢ Hybrid Approach: It is the Hybrid Approach of both Regression Test Selection and Test Case Prioritization. Many algorithms have been proposed for this approach like Wong et. al. [16] technique, Yogesh Singh et. al. [17] technique.

II. A SURVEY ON UML BASED REGRESSION TESTING TECHNIQUES
Test case selection from base test suite is the main task for regression testing. For regression test selection two questions [6] should be taken into account: how to identify the changes in the software, and how to obtain test cases related to those modifications. Regression testing means to select test cases from base test suite as well as to generate test cases for newly added functionalities. A UML Design based technique [18] was proposed to automatically generate the test cases and also to find the affect of change in design. They used class diagrams and sequence diagrams to generate both static test cases and dynamic test cases. Automation is the main point of focus in this
proposed approach but the test cases generated by this approach are represented as sequences of actions which can not be directly executed on the software under test and heavy human effort are needed to transform them into acceptable test cases by software for execution. In [22], a technique of regression test selection based on behavior model is proposed. This technique maps activity diagram to the model used for regression test analysis and selects test cases. But it does not define the traceability between the model and concrete test cases properly. To overcome the above problem Nan Ye et al. [6] proposed an approach to automatically generate executable test cases for regression testing based on activity diagrams. They used regression test case classification technique to select retestable test cases and feedback-directed test cases generation with classifiers(C-FT) [19] to generate test cases to cover the new paths. A tool support is also provided to handle traceability between the concrete test cases and the model.

There are many regression test selection techniques [20] given as: Minimization techniques, Dataflow techniques, Safe techniques, Random Techniques, Retest-All technique.

- **Minimization Techniques**: Minimization based regression test selection techniques selects minimal sets of test cases from base test suite, that covers the modified and affected portions of software
- **Dataflow Techniques**: Dataflow-coverage based regression test selection techniques selects the test cases that exercise data interactions that have been affected by modifications
- **Safe Techniques**: Safe regression test selection techniques select all the test cases that can reveal faults in modified software
- **Random Techniques**: These techniques randomly select a predetermined number of test cases from base test suite
- **Retest-All Technique**: Retest-All technique reuses all the existing test cases

Safe techniques are best to use because safe regression testing techniques guarantee that selected subset T’, contains all test cases in the original test suite T that can reveal faults in P’. Briand et.al. [4] proposed a safe technique for regression testing. This technique considers the dependencies between model elements and change identifications are highly focused in this technique. But this technique focuses on code based test cases which implies that the tester would have to wait for the code to develop and then test it using code based test cases.

Identification of changes is main issue in regression testing and some techniques do not provide sound change definitions. An approach [21] use sequence and class diagrams to detect changes but this approach does not provide sound change definitions. It does not take into account the pre and post conditions of the operations of sequence diagram which affect behavior of a class. So this approach can not select all the test cases according to the changes because they do not consider all the changes in the diagrams and thus it is not a safe technique for regression testing. To solve issue of change definition issue a technique [9] was proposed based on UML class diagrams and sequence diagrams. An Extended Concurrent Control Flow Graph is constructed for both the original system and the modified system and comparator is used to perform node-wise and arc-wise comparison to list all modified and new nodes and arcs. This technique considers class attributes and method constraints and their changes. It also supports concurrency but it did not evaluate the effectiveness of fault-detection precisely.

Modifications in the software may be: addition of some functionality, deletion of functionalities, changes in conditions and constraints etc. Hence regression testing means not only to select test cases from base test suite but it also involves generation of new test cases for newly added functionality. Some of the proposed approach [4, 7] etc. do not provide any information about generation of new test cases for regression testing. The technique [9] identifies the new node for which new test cases are generated but it does not properly generate the test cases.

As for effective regression testing, interdependencies among UML models should be handled carefully. A change in one artifact may cause a change in another artifact without even being reflected on it. For example a change in any operation of class diagram may cause a change in operations of sequence diagram and this change may not be reflected directly in sequence diagram. If this type of interdependencies is not considered then all fault revealing test cases for regression testing can not be selected. A State-based technique [23] has been proposed to handle the dependency between class diagram (structural view) and state machines (behavioral view). In this technique they considers the class driven changes (obtained by comparing baseline and delta version of class diagram) and state driven changes (obtained by comparing baseline and delta version of state machine and by using set of class driven changes). Test cases are classified on the basis of state driven changes. This approach is high in safety because it provides proper change definitions but it is not complete automatic and also does not support any tool to demonstrate the applicability the work.

A regression testing tool START (STAte based Regression Testing Tool) [7] was implemented using the concept of state-based regression testing approach [23]. They also do the performance evaluation of their approach on a case study. This approach shows a significant reduction in size of baseline test suite for regression testing resulting in reduction in testing time and cost.

217
III. CONCLUSION

Regression testing is basically retesting of the software after modifications to ensure that no new errors have been introduced into the previously tested code. Test case selection from base test suite is the main task for regression testing. Regression testing means to select test cases from base test suite as well as to generate test cases for newly added functionalities. Many techniques have been proposed for regression testing using UML models. This paper surveys the regression testing techniques based on UML Models. Unified Modeling Language (UML) Diagrams are best to use for generating and selecting test cases for regression testing as UML design based regression testing techniques have many advantages [11] over code based regression testing techniques like identification of change is easily traceable from design rather than the code, less complexity, understandable etc. UML design based techniques also have many approaches to find out change information of UML designs. As discussed in above techniques, some of the techniques do not provide tool support. Only half of the techniques identify obsolete test cases from the baseline test suite and identify those test cases that need to be rerun to reveal regression faults. Other techniques do not provide proper classification of baseline test suite. Most of the techniques are not safe because they do not provide proper change definitions and also ignores certain elements of models. There is a serious lack of such automatic model based regression testing techniques that follow standard UML versions, identify change properly, classify baseline test suite properly, provides information about added functionalities for which new test cases need to be generated and also provide tool support.

IV. FUTURE DIRECTIONS

One of the future directions is to implement an automatic model that follow standard UML versions and identify changes properly. Another direction is to consider test case minimization, cost factor, test case prioritization, systematic revalidation of UML models along with classification of test cases as these are important during regression testing.

REFERENCES


