CHAPTER 4

A NOVEL METHOD FOR REDUCTION OF REGRESSION TESTING COST

4.1 INTRODUCTION

The computed cost of software maintenance involves to the extent of two-thirds of the average cost of software development. Whenever an update or change in a software application is done, there are possibilities that the other areas within the application to be impacted by this change. Regression testing is performed to check that a fixed defect has not resulted in any other business rule or functionality violation. The aim of regression testing is to make sure that a change such as defect fix, should not result in any new fault that is did not exist in the application. As regression testing is performed repeatedly in software maintenance, it represents the big piece of the maintenance costs (Rothermel. G et al, 2001). Many different approached have been proposed to manage the regression testing efforts. Regression test suite processes, select a subset of existing test set which is available for execution based on some criteria or changes made to the software. Throughout the software maintenance, testing clusters need to execute the regression test set on various requirements, to make sure that the defect got fixed or the upgrades does not impact the other features or functionalities of software. In this study, a methodology to lessen the number of regression test cases in new environment without influencing the scope of the original software. Prioritization of regression test suite helps to find the key risks test cases and executing them first would help to find the limitations before thorough testing.
Figure 4.1 shows the graphical representation of regression test selection technique.

![Diagram of Regression Testing Selection Techniques]

**Figure 4.1 Regression testing selection techniques**

Test suite reduction methods lessen the test suite by identifying and evacuating additional or repetitive tests. This reduction in the regression test suite will lessen the time and effort needed by the testing teams to execute the regression test suite.

**4.2 BACKGROUND OF THE STUDY**

The large number of current methods discuss about test suite which tests the practicality, anxiety, limit values, and execution of the software. Any diminution in this test suite size, will decrease the testing time and effort. Lot of the experiments on test suites, helps to identify the worth and qualities of the software. The proposed technique concludes the reduced test suite is more efficient when compared to the initial test suite. This lessened test suite covers the same benefits of the software as covered by the initial unreduced test suite.
The regression test determination approach is connected on this lessened test suite, to get the number of test cases decreased in the regression test suite. This minimized regression test suite is comprehensive in terms of defect scope and trustworthy as the initial regression test suite. In this proposed study, it is confirmed that testing for purpose and testing for limit qualities could be done with lessened test suite as these two views might be tested together in the vast majority of the situations. The conditions where these two views could be tested, is likewise denoted with help of the detailed analysis. In this study, testing implies, a single test case covers both of the above mentioned standpoint for a particular circumstance. The proposed study is connected on four ongoing investigations and likewise validated the reduction in effort and time of regression testing.

4.3 PROPOSED METHODOLOGY

The cost of software maintenance surpasses 70% of total software costs, and extensive part of this maintenance cost is dedicated to regression testing. Regression testing is an often times executed set of test cases, so lessening the time of regression testing might help in decreasing time of the software maintenance. The proposed study is demonstrated in two phases. In Phase 1, the reduced test suite is determined by implementing the proposed approach on the initial unreduced test suite (Ananda Rao. A et al, 2011). In Phase 2, the reduced regression test suite is determined by implementing a regression test selection method on the Reduced Test Suite that is determined in the Phase 1.

4.3.1 PHASE 1: DERIVING THE REDUCED TEST

There are lot many cases available with different testing procedures to test complete effectiveness of a software system. The test suite consist of test cases for testing stress, limit values, and execution time of the software application. Most of these experiments are testing the usefulness and limit values. The Phase-
1 of this proposed methodology is focused on lessening test cases which still solves the test purpose and limit values. The four main thing before start with the test reduction is, assess whether it is really in limit and worth to test the test suite, recognize the situations where it can be tested with single test case, demonstrating legitimately that the single test case can cover both the viewpoints, applying the above three steps to research with approvals.

4.3.2 PHASE 2: DERIVING THE REDUCED REGRESSION TEST SUITE

Regression test suit creation is selecting a subset of the test cases from the unreduced original test suite. If needed, few new test cases can be also added to test the updated software. Let I is the original software, I’ is the updated version of software and T is the set of test cases to test I. A regression testing for the updated software is as follows. Select I’ an equal or sub set of test cases from I in order to execute on update software system. Test T’ with I’, to verify the updated software’s accuracy with respect to I’. If needed, create T’’ a new test cases to test I’. Test I’ with T’’, in order to verify I’ accuracy with respect to T’’.

In Phase 1, the reduced test suite is determined. In Phase 2, the reduced regression test suite is determined by applying the regression selection strategy. Select a subset of tests from the reduced test suite (determined in Phase1) which covers the essential functionality of the software. Select test cases that covers the criteria to test the issue fixes. Make new tests, to test the new upgrades included in the regression test suite.
Figure 4.2 shows the graphical representation of proposed framework for this research work.

![Proposed Framework Diagram](image)

**Figure 4.2 Proposed Frame work**

In step 1 of this methodology, the subset of test suite is selected from the original test suite. By this, the chosen subset also has the less number of tests than the original unreduced test suite. This reduced regression test suite covers the same intention as the original unreduced regression test suite that is generated without applying this proposed method.

### 4.4 RESULTS AND DISCUSSION

The proposed approach is applied on MYSQL ETL DB and SYBASE ETL DB mechanism of a data warehousing tool. The results shows the comparison of size of the test suite and time taken to execute the original test suite and reduced regression test suite by the proposed method.
The results in table 4.1 shows that the proposed approach saves significant amount of regression testing effort and time.

### Table 4.1 Reduction in Regression Testing

<table>
<thead>
<tr>
<th>DB Component</th>
<th>Original Regression test suite</th>
<th>Reduced Regression test suite</th>
<th>Time for original regression test suite</th>
<th>Time for reduced regression test suite</th>
<th>Percentage of reduced regression testing time</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYBASE ETL DB component</td>
<td>1400</td>
<td>1000</td>
<td>3500</td>
<td>2600</td>
<td>24.33%</td>
</tr>
<tr>
<td>MYSQL ETL DB component</td>
<td>1600</td>
<td>1100</td>
<td>2998</td>
<td>2011</td>
<td>18.39%</td>
</tr>
</tbody>
</table>

Table 4.1 indicates that by employing proposed method the reduction in regression testing is 24.33% and 18.39%.

### 4.5 SUMMARY

The proposed methodology reduces the amount of regression testing effort and time, independent of the regression test selection schedules that are accessible. The exertion needed to apply this method is one time effort, yet it lessens the effort and time required for all remaining regression cycles of the software. Hence with the proposed approach the regression testing effort can be reduced.