International Journal of Electrical Electronics & Computer Science Engineering
Special Issue - ICSCAAIT-2018 | E-ISSN : 2348-2273 | P-ISSN : 2454-1222
Available Online at www.ijeecse.com

A Study of Effective Regression Testing

Nisha Jha
Assistant Professor, Department of Computer Science, Lingaya's University, Faridabad, Haryana, India

***Abstract:** Software Quality is one of the major challenges faced during the growing needs of business and IT infrastructure. Software testing has the major impact on the software quality. However, it is quite an expensive task because it requires a lot of effort and time. Regression testing is one of the approaches of software testing that ensures that the changes in the functionalities of the system will not affect the existing functionalities of the software.*

Regression means when a bug is fixed, several other bugs are introduced. It can be viewed as the extension to the retesting which does not focus only on the defected module but also the inter-related modules to ensure error free software.

Regression testing involves the cost of retesting the previously testing components iteratively, which increases the importance of efficient regression testing strategies. Researchers have proposed several techniques to optimize the regression test cases. The paper focuses on the different regression testing approaches and the challenges faced by the testers during test case optimization.

***Keywords:** Regression Testing; Test Case Selection; Test Case Minimization; Test Case Prioritization; Software Testing; Retesting.*

I. INTRODUCTION

It has been proved that complete testing is not feasible or practical .No matter how well executed and tested software is, it will definitely have some bugs .Software will surely have to be modified as per dynamic user specification or requirements, or to fix bugs .These modifications may introduce some new bugs or may affect some of the existing features of the system [1].So it becomes important to test the whole software again so that a new modification does not affect the certain part or other modules of the software.

Regression testing is conducted to check that changes made to software, like adding new features (new data flow paths are established, new I/O may occur), have not affected the other functionalities of the software [2].

To reduce the stress levels of development teams, various automated testing tools that are specialized in regression testing make it relatively easy, with a few clicks of a mouse, to set testing parameters and to check new iterations of code against previous software baselines, or control states, highlighting inconsistencies in testing logs and specifying exactly where an unexpected function occurred and why.

Testers may rerun all the test cases at the earlier phases of Software testing life cycle (STLC) to verify whether the program behaves as expected. However, when the program evolves during maintenance phase, the set of regression test cases grow larger. Repeating all previous test cases along with new test cases after each minor change in software is very tedious task as well as very time consuming. So the regression testing is expensive but definitely necessary task. Researchers are mainly focusing on how to reduce such cost [2].

During maintenance phase, the software is modified to fix defects, change functionalities and satisfy new requirements with respect to specifications and implementation. For both type of changes regression testing can be divided into two types: Corrective regression testing and Progressive regression testing.

Corrective regression testing can be used when there is no change in the specifications and test cases can be reused. Progressive regression testing is used when the modifications are done in the specifications and new test cases are designed [3]. Fig. 1 shows the various activities during regression testing.

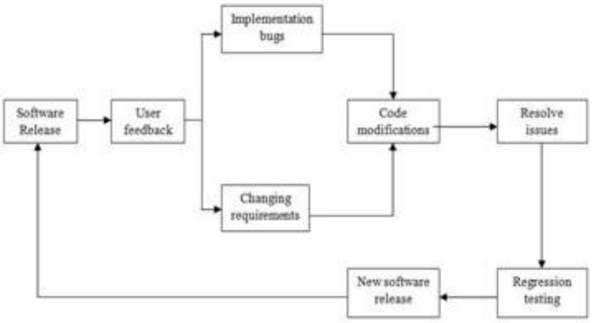

Fig. 1. Activities During Regression Testing

Difference Between Re-Testing and Regression Testing is as Follows:

Retesting is performed by the tester to check that the defects reported in the previous build have been fixed, which requires re-running the failed test cases. Consider a scenario where a build is found to have 10 defects, which are then assigned to developers for fixing. Once all these defects are fixed, the new modified build is retested to verify that the defects have really been fixed, which is why it is called as retesting.

Regression Testing is performed to identify the new bugs that are introduced due to fixing of the previously reported bugs or changes made in the previous build. Here, it is checked that the bugs reported in earlier build have been fixed, which means retesting the given build. Regression testing is accomplished through the construction, execution and analysis of system tests. The main objective of regression testing is to ensure that the changes introduced into build are correct and do not adversely affect the other system components.

There can be a couple of cycles for the regression testing that can be deliberately for every release. This can be as a result of bug fixes which may come in phases or some bug fixes may not work as expected leading to extra regression cycles.

II. Regression Testing Techniques

Many techniques have been devised on how to select the regression test cases so that number of test cases doesn't increase too large and cost and effort can be minimized. Figure 2 shows the various techniques for regression techniques [4].

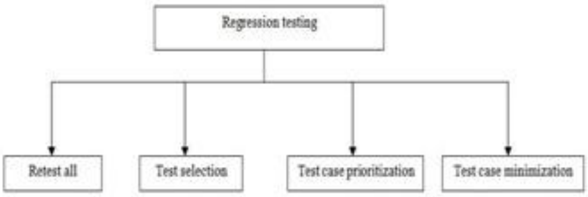

Fig. 2. Regression Testing Techniques

A. Retest All:

In this method, all the test cases are re-executed. This approach is costly to execute as it require more time, budget and resources. Because here all the test cases are reused this results in the execution of unnecessary test cases. When any small modification or change is done to the application then this strategy is not useful [5]. The retest-all approach is considered impractical on account of cost, resource and delivery schedule constraints that projects are frequently subjected to.

B. Regression Testing Test Selection:

This method reduces the cost of regression testing by selecting some subset of an existing test suite to retest the modified program. The approaches or techniques that are most often used in the industry for the detection of regression test cases ,which are either based on expert judgment, or based on manual program analysis [6]. Even for moderately complex systems, it is generally difficult to manually detect test cases which are relevant to an amendment .This approach often leads to the selection of large number of test cases and rerun all the test cases even for small changes in the original program, leading to unnecessarily high regression testing costs [7].

A variety of regression test selection has been described by different authors in research field:

Safe Techniques: This technique does not focus on criteria of coverage but select all those test cases that produce different output with a modified program as compared to its original version.

This technique by definition eliminates only those test cases that are probably not able to reveal faults [8].

Minimization Techniques: This technique aims to reduce the size of test suite by removing the redundant test cases.

Data Flow Technique: This technique is coverage-based regression test selection technique that selects test cases that exercise data interactions that have been affected by modifications.

Retest All technique: This method reruns all test cases in T that were previously run during testing phase and not a single test case is left [9].

Adhoc/Random Technique: When time constraints are there and no tool is available, this technique allows developers to randomly select the test cases based on their experience or intuitions [10].

C. Test Case Minimization:

It reduces the test cases by eliminating the redundant test cases from test suites in terms of code and functionalities. A redundant test case is one which gives the same output as other test case [11]. But any type of minimization is risky and may omit some fault revealing test cases.

Effective Regression Testing may be performed by selecting following test cases:

- Test cases which covers frequent defects.
- Functionalities or features which are more visible to the users.
- Test cases that verify the core functionalities of the software.
- Test cases or the part of the module which has undergone most changes recently.
- Test Cases related to the integration testing.
- Complex Test Cases.
- Test cases based on the boundary value analysis.

D. Test Case Priortization:

In this technique, the testers prioritize the regression test cases based on various important factors such as total efforts, deploying time, execution time, cost-benefit tradeoffs etc to increase the fault detection rate of a test suite. This technique reorders test cases and schedules the test cases with the higher priority on the basis of some criterion decided earlier in the testing process using some random technique. By prioritizing the execution of a regression test suite, these methods reveal some critical defects in software in the regression testing [12]. The main motive behind prioritizing the execution of a regression test suite is to ensure that the defects introduced or discovered by competent developers are discussed at earlier stages in the testing process.

In this method, the important point is to ensure that the severity of low-priority test cases should not be high. Prioritize the test cases relying on business impact, crucial and often used functionalities. Selecting the test cases on the basis of priority will significantly lessen the regression test suite.

Researchers have described several techniques for prioritizing test cases. Their results show that test case prioritization can significantly improve the rate of fault detection of test suites. Some of the techniques for test case prioritization are:

Total Statement Coverage Prioritization: It ranks the test cases based on the total number of test cases covered.

Additional Statement Coverage Prioritization: It can execute those statements also which have not been executed in any previous test cases.

Total Branch Coverage Prioritization: The criterion, here, to order is condition branches in the program.

Additional Branch Coverage Prioritization: It selects those test cases which covers the uncovered branch outcomes.

Total Fault Exposing Potential Prioritization: Fault exposing potential can be defined as the ability of a test case to find a fault. It orders the test cases based upon the criterion that which test cases cover a faulty statement and which fault in that statement can cause a failure.

Risk Based Prioritization: It uses risk analysis for prioritizing modules for testing.

Prioritization Based on Requirements: It is dependent on the requirement analysis for prioritizing the test cases.

Test case prioritization can address a wide variety of objectives, as given below [15]:

- Intention of testers to increase the fault detection rate.
- Finding the high-risk faults at the earlier stages of the testing life cycle.
- To increase the code coverage at a faster rate.
- To increase the reliability of the system.

III. Regression Testing Tools

Here, some of the best commercial regression testing tools has been briefly discussed:

Q Engine: QEngine offers integrated solutions to test and verify the functionalities and performance of web applications. It provides QEngine toolbar for remote and multi-user record/playback and provides web-based and data driven test script creation and maintenance. There is a QEngine Issue Manager to track and manage issues.

Selenium IDE: It is an open source software tool used to test web base applications, implemented as a Firefox extension. It is used to record, edit, and debug tests. Selenium IDE is not just a recording tool but it is a complete IDE. You can choose to use its recording capability, or you may edit your scripts by hand. The selenium testing is not just a single testing tool but it is a suite of different tools which are Selenium IDE, Selenium Web Driver, Selenium RC, and Selenium Grid. With auto complete support and the ability to move commands around quickly, Selenium IDE is the ideal environment for creating Selenium tests no matter what style of tests you prefer. It supports multiple languages like java, c#, and python to create tests.

QA Wizard: QA Wizard Pro automates the functional and regression testing of Web and Windows applications. It provides more testing suite in less time. It is a powerful and Data-driven Testing tool which provides validation checkpoints and Global Application Repository. It is an easy to use scripting language.

TestDrive: TestDrive-Gold is used to test any GUI or browser application ‘out of the box’. It has in-built technology which easily creates test scenarios through a simple point-and-click interface.

It has the ability to execute a complete regression test in a very less time with complete results.

Rational Functional Tester: It provides automated testing capabilities for functional testing, regression testing, and GUI testing and data-driven testing. It is code free testing tool. It also supports .NET, Java, Ajax, Oracle, PHP etc. It reduces time by increasing test coverage and achieves agile automation.

JUNIT: JUNIT has emerged as an important test tool. It provides unit testing tool framework for Java programming language.

IV. Search Algorithms For Test Case Prioritization

There are many search algorithms for test case prioritization being developed by many researchers [13]:

A. Greedy Algorithm: Greedy Search seeks to minimize the estimated cost to produce the quality results as it is least expensive in terms of both implementation and execution.

B. Additional Greedy Algorithm: It iteratively selects the max_weight element from that part of the program which has not been consumed by previously selected elements.

C. Genetic Algorithm: It is a population based search method which uses several parameters like mutation, cross-over, and population size.

D. Artificial Bee Colony: It is a swarm intelligence algorithm which was developed by Karaboga in 2005 and stimulated with the aid of the behavior of the honey bees. It is developed based on inspecting the behaviors of real bees on finding the nectar and sharing the knowledge of food resources to the bees within the hive [16].

E. Hybrid Technique: It is a combined approach of regression test selection and test case prioritization.

V. Advantages And Challenges Of Regression Testing And Test Case Prioritization

A. *When to use Regression Testing :*

- When a new feature is added or enhancement is done
- A bug has been fixed
- Any performance related problem has been solved

B. *Advantages:*

- It helps us to make certain that any changes like fixing bugs or any improvement done to the module or software have not impacted the prevailing tested code.
- It guarantees that the bugs found earlier aren't creatable.
- Regression testing can be automated using the automation tools to reduce effort and time.
- It helps in raising the standard of the product.

C. *Disadvantages:*

- It is very time consuming and expensive task to repeat all the previous test cases along with the new test cases after each minor modification is done.
- Sometimes test case minimization may be risky because it may remove some fault revealing test cases.
- Test selection technique requires expert judgment and a lot of manual work and analysis.

- Considering only one criterion for test case prioritization may not significantly improve the early fault detection like the test case which covered maximum coverage is given higher priority may not cover the fault revealing case.
- If different automation tools are used then the test prioritization results may vary because they use different criteria like blocks, methods or classes for measuring coverage.
- In an agile environment, where resources are limited and changes are very frequent; regression testing may introduce extra overhead [14].

VI. Conclusion

Regression Testing is a costly and tedious but crucial activity. The purpose of regression testing is to achieve quality product with least cost and time. Many researchers are exploring this field of testing and have reported many techniques to minimize the cost and effort by test case prioritizing algorithms, and creating various test selection tools for minimizing the test cases. This paper is an initial work to analyze both, positive and negative sides of regression testing. The main idea of this review is to discuss the different test case optimization techniques with their importance.

VII. References

[1] R. Pressman. Software Engineering: A Practitioner's Approach. McGraw-Hill, New York, 2002.

[2] H. Leung and L. White, Insights into regression testing. In Proceedings of the Conference on Software Maintenance, Oct. 1989,60-69.

[3] Jyoti, Kamna Solanki, A Comparative Study of Five Regression Testing Techniques : A Survey, International Journal of Scientific & Technology Research, Volume 3, issue 8, august 2014.

[4] G. Baradhi and N. Mansour. A comparative study of five regression testing algorithms. In Proceedings of Australian Software Engineering Conference, Sydney, pages 174–182, 1997.

[5] Chandana Bharati1 and Shradha Verma,- Analysis of Different Regression Testing Approaches, International Journal of Advanced Research in Computer and Communication Engineering,Vol. 2, Issue 5, May 2013.

[6] G. Rothermel and M. Harrold. A safe, efficient algorithm for regression test selection. In Proceedings of the Conference on Software Maintenance, pages 358–367, 1993.

[7] Sunita, Ms Mamta Gulia, "Study of Regression Test Selection Technique", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 2, February 2014.

[8] E. Engström, P. Runeson, and M. Skoglund. "A systematic review on regression test selection techniques", Information and Software Technology, 52(1):14–30, January 2010.

[9] P. Chittimalli and M. Harrold. "Regression test selection on system requirements", In ISEC '08: Proceedings of the 1st conference on India software engineering conference, pages 87–96, 2008.

[10] G. Rothermel and M. J. Harrold, Analyzing Regression Test,Selection Techniques, IEEE Transactions on Software Engineering, V.22, no. 8, August 1996, pages 529-551.

[11] Shin Yoo, Mark Harman,"Regression Testing Minimisation, Selection and Prioritisation : A Survey", pages1-6.

[12] Zengkai Ma, Jianjun Zhao, "Test Case Prioritization based on Analysis of Program Structure", 3-5 Dec. 2008.

[13] Zheng Li, Mark Harman, and Robert M. Hierons, "Search algorithms for regression test case prioritization", 4, April 2007**.**

[14] https://en.wikipedia.org/wiki/Regression_testing

[15] Ahlam Ansaria, Anam Khanb, Alisha Khanc, Konain Mukadamd. “Optimized Regression Test using Test Case Prioritization”, 7th International Conference on Communication, Computing and Virtualization 2016, 79 (2016) 152 – 160.

[16] Gurcharan Kaur1, Bhupender Yadav. “Test Suite optimization Using Artificial Bee Colonyand Adaptive Neural Fuzzy Inference System”, International Journal of Science and Research, Volume 4, Issue 10, October 2015

.