Research Article

Regression test optimization and automation in Agile framework: A Review

Vivek Kulkarni[a], Dr. R. Madhan Mohan[b], Dr. H. Venkateswara Reddy[c]

[a,c] Vardhaman College of Engineering Hyderabad, India
[b] Department of Computer Science and Engineering, Annamalai University, India
[a] vivek@vardhaman.org, [b] madhanmohan_mithu@yahoo.com, [c] h.venkateswarareddy@vardhaman.org

Article History: Received: 11 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 23 May 2021

Abstract: Regression testing is a well-established practice in software development and has shown itself to be one of the best ways to test the software before its release. The modified code and some existing code affected should be re-tested during regression testing. This often takes time and becomes painful. Therefore, test automation is increasingly required and important in software engineering to carry out the regression test.It is very obvious that nowadays, many companies invest in automated testing tools to prevent defects while developing. Agile methodologies are one of the automated regression testing tools available. The major advantages of agile software development are delivery and change in high-quality software in shorter intervals. Test strategies are well established for conventional process models, but these strategies are not directly applicable without changes or changes to agile tests. Agile methodologies although useful in automation regression testing,have seen some challenges for regression testing. This paper presents detailed examinations and research evidence using agile methods and test automation. It will also address optimization techniques in agile methodologies and challenges faced in agile.

Keywords: Regression testing, Test automation, Testing tool, agile methodologies, agile software development

1. Introduction

Software testing takes a long time to operate for testing the project and can be the costliest step of the software development process. The checked part of the software development process is understood to be the least. Also, machine experiments are carried out repeatedly and quite rapidly because of time limitations and budget constraints. Regression testing is an essential form of software test, which mostly guarantees and verifies the effect of any new code on the product's current features.

This testing form helps testers to detect faults or errors in the build as code modifications are made and assure that all functionality and functionalities are stable even though code modifications are made. Regression testing is essentially a black box evaluation procedure that guarantees regular code updates that do not compromise the functionality of the device. Regression checking on a new build can be carried out because the initial functionality has drastically improved, even with a single bug patch. It is typically done after improvements have been reviewed or new functions are introduced and any new function should be regularly evaluated.

The agile growth framework for automation testing takes hold, with possible benefits including increased client collaboration, increased colloquially and pairing encounters. Besides, tests for agile development are becoming a central component and are being needed by most software testing teams, particularly when the time for overall evaluations like regression tests is restricted due to short periods between gradual software upgrades. Agile testing in automation would also be very useful and advantageous in regression testing.

In this paper, we are focussing on regression testing and agile methodologies for regression testing. The rest of the paper is organized in the following sections. Section 2 focuses on the literature review and background work of regression testing and agile methodologies. Section 3 introduces the regression part in an agile development environment. Section 4 focusses on basic concepts of agile in regression testing. Section 5 discusses the implantation of agile testing in regression. Section 6 points out the optimization process in agile methodologies. Challenges in agile projects are discussed in section 7 and section 8 concludes our finding and future directions.

2. Related Work:

Regression Testing:

Regression testing is an essential form of testing that, even after alteration to the program, is responsible for the stability, consistency, usability and functionality of current applications. Regression checking is an operation that guarantees that the change is not obstructed before software operates. Changes are designed and are an essential part of the software development life cycle. A test suite of applications expands and continues to expand, making it expensive to run as well.Analysis has shown that regression testing is an expensive process that can claim over 33 percent of the total cost of the software[1]. Regression checking is an expensive but important testing operation that ensures that code changes have not already modified functions in use [2]. The full test suite is an unfeasible job after any code change because of time constraints.

Various methods and techniques for test case reduction and test case priority testing, including manual and automation techniques, consist primarily of greedy algorithms, genetic algorithms and a whale-optimization algorithm were proposed in previous studies [3][4][5][6].

Khatibsyarbini M. et al.[7] investigated a distance-based hybrid string similarity method in association with the PSO algorithm (PSO). The distance calculated by means in terms of sequence arrangements in a set. The significance and discrepancy of the two strings are seen in a string distance. In this way, test case information can be translated into string formats to measure the similarities and differences between test cases. The method presented uses four-string matching functions: index manhattan, Levenshtein, cosine similitude and index Jaccard. All four string matching functions have been hybridized by the author. As a data collection for approach validation, Siemens SIR programs repository will be included. The validation is based on the APFD metric demonstrating a superior solution than non-hybrid string matching.

To create test restrictions on any problem found before, Chang-ai Sun[8] has implemented an additional test suite reduction approach for conservative regression testing in the Boolean format, which lays out the criteria that might ensure identification of the targeted bug within a testing scheme.

The priority of the test case is to strengthen the regression testing by re-programming the test cases to increase the rate of fault detection. In the field of software testing, risk-based testing became more common. However, most new methods manually measure the risk values, which exhausted, laborious and sluggish these methods [9][10].

The algorithm Cuckoo Search (CS) and Modified Ant Colony Optimization (M-ACO) were combined to finish the trials in a time-restricted order. The entire test suites, however, are difficult to manage due to cost and time constraints[11].

2.2 Agile Development idea:

Automated regression testing makes the testing even more effective, as manual testing tediously constantly takes place. Automation can free up time that can be used to minimize tedious repetitive tasks when using this automation process.

The Agile Manifesto was coined in February 2001 by experienced and leading software professionals[12]. At the Snowbird, Utah Resort, software developers met to explore lightweight development methods. They also wrote the Agile Software Growth Manifesto that outlines the most important points:

Individuals and Interactions – Self-organisation, inspiration and interactions such as colocation and pair programming are crucial for agile development.

Working Software – Working technology is more valuable and valued than merely delivering customer documentation at meetings.

Customer Collaboration – At the outset of the software development period, specifications cannot be entirely collected; constant consumer or stakeholder engagement is thus very necessary.

Responding to Changes – Fast responses to transition and continuous improvement are the focus of agile development. There are some relevant concepts to adopt in the Manifesto during the software development process, including checking. The emphasis is on quick, iterative product delivery.

In short, agile software is an on-going, gradual growth-focused community of software development approaches. The specifications and strategies in this setting evolve by working with cross-functional self-organising teams. Agile fosters agile preparation, evolution and implementation, a time-consuming iterative approach and facilitates fast and versatile change response. This is a conceptual paradigm that promotes tight relationships across the cycle of creation. Some traditional and some new creation methods, including XP and Scrum, are close to agile. For these approaches, certain agile concepts are conflicting principles. However, in this essay we will continue to concentrate on agile, it should be noted that certain test elements still relate to these methods in several respects.

3. Regression Testing

Regression testing is the main class of tests conducted to ensure that changes/changes/upgrades in the features are not influenced by current functions. The regression research is continuing, as 70 percent of SDLC is updated, fixed defects and maintenance were observed [13]. Whenever improvements in code occur through the removal of a fault or where new functions are introduced, certain features are influenced by the relation between them. The procedure for regression is different from the test for the retest. Whenever a defect has been re-tested, the defect has been entirely corrected and the defect has been closely resolved. Retest meant that the test cases struggled again.Regression testing is entirely different from re-testing. It seeks to evaluate and ensure that all functionalities previously operated are now constant and that even after improvements to the program, they execute perfectly. For selective failure scenarios, regression testing is a re-testing technique. Re-testing is completed.

When the application is modified, it undergoes regression testing to make sure that no new faults have occurred in the earlier tested functionalities of the application under test (AUT) and it still works acceptably [2][14]. figure 3.1 depicts this concept of regression testing.

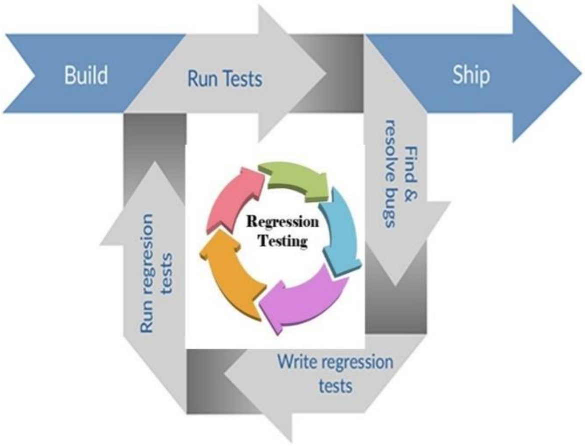

Figure 3.1: Regression Testing Process Source [13]

4. Regression Testing inan Agile Context: Basic Concepts

Regression checking of agile guarantees company processes is continued for the accelerated program improvements. In the agile background, the team helps to concentrate on designing new features and general functionality in the sprint. Automation monitoring and continuous output help to gain successful agile regression tests.

Regression testing in agile helps keep the program steady with each new product rise. Regression checking for improved software consistency is used in agile. With regression testing, the current product functions are tested and no new code updates are affected. Ses adjustments can improve, repair or alter the product's configuration or more. Regression testing helps to provide free tools for bug distribution.

Checking regression means that no new improvements in the program are made. It may be used for any improvements in the program located or wider. Agile regression testing is very critical as these regular builds of agile are constantly altered in application and usage. A research team can develop regression test suites in addition to the development process for efficient regression testing in agile.

Regression Testing in agile is done by several types of regression testing. The method one prefers for regression depends on the coverage of the test, the availability of testing tools and the priority of an evaluated program.

5. Implementing Regression Testing In Agile

It is necessary to have a product that is not only well checked but also properly evaluated for regressions to gain confidence in the consistency of the product being supplied. Implementing regression checks in agile guarantees the general product consistency, even with new characteristics introduced per sprint, is what the consumer wishes.

Regression checking is carried out in agile applications to create overall reliability. Regulation testing in agile at the desired time eliminates the programming rework for products that, due to insufficient testing, do not work as intended.

As manual regression testing in agile is a time-consuming process, it is easier to automate and implement regression testing suites, which ultimately saves time and also improves test coverage. With several automation tools available on the market, the automation of the control suites needs to choose the correct automation tool. Automation platforms deliver uncoded automation checking.

In any sprint where regression tests are needed, the automatic test cases created for regression tests can be reused, saving a great deal of time.

We do need to train for regression in agile as soon as possible before the development process stops, during the introduction of regression checks. Regression testing should be begun after production has been finished as early as possible. This lets us stick to the timetable.

When the regression test is automated, regression testing saves the time that testers expend and the test team can concentrate on covering more parts of the program, and run other kinds of tests other than regression. Otherwise, it could help to get published sooner.

With the increasing focus given to product consistency, the use of regression checks in agile means that any single timeline is consistent at the end of the sprint.

6. Optimizing Regression Testing in Agile Development

To maximize regression testing in Agile methodology, attempts must be made from the beginning of the development:

- The testing team must collaborate seamlessly with both parties. The testing team must collaborate with engineers, market analysts and product owners long before the regression period starts to optimize the new functionality. Centred from a test perspective, market analysts will track improvements to the requirement.
- The research team can develop test cases based on the application priority while closing the requirement gaps. The related test cases can be transferred in Regression Test Suites until the characteristics are consistent. This test cases may be broken down into cases of extreme, moderate and low priority. Ensure that the risks are protected by the architecture of the agile regression test suite.
- Automating the test suite reduces time and effort before conducting the test. So teams should engage in the automation of regression testing after evaluating all the efforts involved and the outcomes if the ROI is optimistic.
- Document a regression test strategy identifying testing range, testing methodology, test methods, test forms (unit/integration/acceptance) of test environmental data, test entry and exit conditions, contingency plan probability risk, running schedule, the resources available, the tool for deficiency monitoring, etc.
- Observe a daily peer review for the test cases built.
- Selecting high-priority test cases for execution when the specified time is reduced would increase test coverage and improved the detection of bugs with less time.
- Engage in the agile refinement session for all involved parties and schedule progress on test cases.
- Stick to the summary comments and make appropriate adjustments in the test cases where necessary.
- Schedule preparation for feature functionality and the automated tool before the start of the regression.
- Emphasis on direct dialogue over the everyday agile stands to elevate every challenge to the testing process. Keep commodity owners aware of the present state of regression.

7. Regression testing challenges in Agile projects

There is a chance for teams to face something like a challenge when regression testing is used in Agile programs. For eg, there are a variety of unique automation problems here.

- Maintenance - Automatic test cases for an unspecified time cannot and will not be valid. That is to say, maintenance is important. In particular, testers must upgrade and adjust automatic test cases based on project enhancements and updates. Simply placed, redundant test cases must be deleted.
- False-positive bug findings regression testing but nothing ever goes wrong with the software. You play a game of locating a guilty guy in the water. It could be difficult and time-consuming to locate the root of this problem. The defect will vary from a time problem to a badly coded test case.

In addition to automation, additional challenges can emerge during regression testing. Here are some of the most popular ones.

- Repeated changes: consumers, partners or managers make changes and wipe out all of them. Any automated technique for the regression test is now at risk.

- No recording or replay review software – You need to wait until the application is ready to use a conventional research tool for recording or playback capabilities. In other words, standard automated testing in an agile environment cannot be used.

- Measure growth - regression testing improves considerably with any run, to the degree that it cannot be performed more rapidly. Therefore, experiments must try to maintain the test at a manageable pace by the regular optimisation of test cases and the elimination of redundant cases.

- Ineffective communication - Collaboration and communication are thriving on agile teams. Unproven improvements in tech, which may cause grave problems in the future, could result in some changes without effective contact between tester, developers, market analysts or stakeholders.

- Advanced testing ability - the more special testing capabilities are required for a project. This refers in particular to efficiency and integration monitoring. The team should either have a specialized tester from the outset or at least one inside the organisation who can help if appropriate.

- Managing the test cases - Automation allows to validate functionality consistency with the automation of further test cases. More automation, however, demands more maintenance. Make a test too loose and a bug can occur. On the other hand, make the test strict and for any slight adjustment, you will need to correct it.

8. Conclusion:

Regression testing and automation of that using agile methodology had gained importance in today's scenario of the software testing world as the time constraint for testing the test cases with prioritization are very crucial. In this paper, we had focussed on the regression testing and agile methodologies useful in regression testing. We had also discussed optimisation techniques possible in agile methodology and challenges faced by agile methodologies. In future, we will be working on different optimization techniques and will apply using agile methodologies for faster execution of test case scenarios.

References

1. P. K. Chittimalli and M.-J. Harrold, "Recomputing Coverage Information to Assist Regression Testing," IEEE Trans. Softw. Eng., vol. 35, no. 4, pp. 452–469, 2009, DOI: 10.1109/tse.2009.4.
2. S. Elbaum, P. Kallakuri, A. Malishevsky, G. Rothermel, and S. Kanduri, "Understanding the effects of changes on the cost-effectiveness of regression testing techniques," Softw. Testing, Verif. Reliab., vol. 13, no. 2, pp. 65–83, 2003, DOI: 10.1002/stvr.263.
3. P. Kandil, S. Moussa, and N. Badr, "A methodology for regression testing reduction and prioritization of agile releases," Mar. 2016, DOI: 10.1109/ICTA.2015.7426903.
4. S. Ghiduk, M. R. Girgis, and E. H. Abd-Elkawy, "A Survey of Regression Testing Techniques," Int. J. Adv. Res. Comput. Sci., vol. 3, no. 5, pp. 140–148, 2012, Accessed: Feb. 21, 2021. [Online]. Available: www.ijarcs.info.
5. Bharati and S. Verma, "Analysis of Different Regression Testing Approaches," 2013. Accessed: Feb. 21, 2021. [Online]. Available: www.ijarcce.com.
6. S. Elbaum, A. G. Malishevsky, and G. Rothermel, "Prioritizing Test Cases for Regression Testing."
7. M. Khatibsyarbini, M. Adham Isa, and D. Norhayati Abang Jawawi, "A HYBRID WEIGHT-BASED AND STRING DISTANCES USING PARTICLE SWARM OPTIMIZATION FOR PRIORITIZING TEST CASES," J. Theor. Appl. Inf. Technol., vol. 30, no. 12, 2017, Accessed: Feb. 17, 2021. [Online]. Available: www.jatit.org.
8. C.-A. Sun, "A Constraint-based Test Suite Reduction Method for Conservative Regression Testing," 2011, doi: 10.4304/jsw.6.2.314-321.
9. H. Jahan, Z. Feng, and S. M. H. Mahmud, "Risk-Based Test Case Prioritization by Correlating System Methods and Their Associated Risks," Arab. J. Sci. Eng., vol. 45, no. 8, pp. 6125–6138, Aug. 2020, DOI: 10.1007/s13369-020-04472-z.
10. G. Rothermel, R. H. Untch, C. Chu, and M. J. Harrold, "Test Case Prioritization: An Empirical Study," 1999.
11. Y. Zhang et al., "A hybrid ant colony and cuckoo search algorithm for route optimization of heating engineering," Energies, vol. 11, no. 10, 2018, DOI: 10.3390/en11102675.
12. "Manifesto for Agile Software Development." https://agilemanifesto.org/ (accessed Feb. 21, 2021).
13. S. Yoo and M. Harman, "Regression testing minimization, selection and prioritization: a survey," Softw. Testing, Verif. Reliab., vol. 22, no. 2, p. n/a-n/a, Mar. 2010, DOI: 10.1002/stvr.430.
14. Kiran, W. H. Butt, M. W. Anwar, F. Azam, and B. Maqbool, "A Comprehensive Investigation of Modern Test Suite Optimization Trends, Tools and Techniques," IEEE Access, vol. 7, pp. 89093–89117, 2019, DOI: 10.1109/access.2019.2926384.