IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

METHODOLOGIES AND APPLICATION OF AUTOMATED TESTING FOR MISSION SOFTWARE ELEMENTS

[1]Savitha. A, [2]Sudeesh B
[1]Scientist Engineer, [2]Group Head
Reliability and Quality Assurance Software Group
UR Rao Satellite Center, ISRO, Bangalore

Abstract: U R Rao Satellite Centre (URSC) of the Indian Space Research Organization develops satellites for variety of applications like remote sensing, interplanetary, communication, navigation and scientific applications. Now the prestigious mission Gaganyaan is planned. These missions consist of highly complex software which will carry out many tasks. Autonomy feature is also built in to handle failure modes also change over to redundant system if one system fails. From one spacecraft to another spacecraft the complexity and the functionality of software is increasing. Hence testing and delivering the robust bug free software is a major challenge within the project time period. Hence realizing this software plays a major role in the success of mission. The automation of testing saves lot of project time and cost. This paper discusses the methodologies and application of automated testing for mission software elements. It discusses the best practices for implementing automation testing. Safety and security play a critical role in every phase of software development life cycle.

Keywords — ALC (Application Lifecycle Management), AUT(Application Under Test), ATLC (Automation Testing Life Cycle), DCL (Data Control Language), DDL(Data Definition Language) DFD (Data Flow Diagram), FDD (Functional Design Document), GUI(Graphical User Interface), IDE (Integrated Development Environment), MST (Mission Scenario Test), QA (Quality Assurance), ROI (Return On Investment), V & V (Verification & Validation), SRS (Software Requirements Specification)

I. INTRODUCTION

The software development team will test the software developed and the QA team tests it again, yet delivered software shall have defects. Test engineers attempt to catch them before the software is operational but still some bugs will appear, even with the best manual testing processes. Automated testing is the best way to reduce the bugs and increase the efficiency, value, code coverage, reduce time and cost Ref [1]. Manual testing is preparing test cases, preparing various input combinations and logging the outputs and observations. It is useful during development with code changes also with multiple operating environment and hardware configurations. Automated testing helps to test multiple times and record the results and compare the results. Test report can be generated for success/failure of test cases tested Ref [2]. Hence test engineer will be more confident on automated testing and providing quality report on the software being delivered. Automation also means reuse and run the test cases many times.

II. AUTOMATION TESTING

Automated testing helps to bring down the development cycle time and helps to improve software quality. It prevents the substantial amount of repetitive task. Implementation of automated testing for mission software elements will be the best preparation and the groundwork for improving software quality. Thorough testing is essential for the proper working of the mission software elements as per the requirement at the user stations. Changes to mission requirements at later stage in development cycle are inevitable. Testing to find anomaly is too expensive and time consuming also a repetitive work which focus to human error.

Automated testing is a comprehensive, cyclic and rigorous tests which is carried out automatically which helps to improve software quality with the limited testing resources. Automated testing helps to test faster, also more code can be tested, test accuracy can be improved also helps the quality engineer to carry out independent verification and validation smoothly. Tests can run automatically even if some source code changes and later compare results. This helps in impact analysis by saving time and increase assurance. Even a good tester will make mistakes in manual testing, automated tests perform the same more precisely, repeatedly and the results are recorded for detailed analysis. The engineer can focus on the manual tests which needs human attended services.

The software tests that can be automated are:

- Unit Testing
- Functional Testing
- Regression Testing
- Black Box Testing
- Integration Testing
- Data Driven Testing
- Smoke Testing

The automation testing is simple. Figure 1 shows the steps to be followed in automation testing.

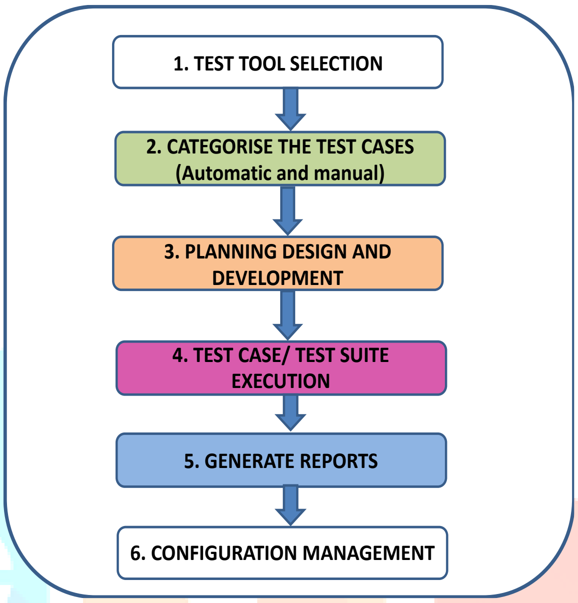
1. TEST TOOL SELECTION
2. CATEGORISE THE TEST CASES (Automatic and manual)
3. PLANNING DESIGN AND DEVELOPMENT
4. TEST CASE/ TEST SUITE EXECUTION
5. GENERATE REPORTS
6. CONFIGURATION MANAGEMENT

Figure 1: Steps in automation testing

III. APPROACH FOR AUTOMATED TESTING

Independent verification and validation need continuous testing staring from unit level testing. Continuous testing is the evolution for test automation in our mission scenario software elements. Because some modules will be modified and used by other project as the new requirements or modified requirements are coming for the upcoming spacecrafts with different sensors and payloads. Hence some mission software's are multi mission software's which needs continuous testing that in turn leads to advancement of systematized testing. All the key qualities of testing like functional, security, performance, boundary tests etc., shall be fully automated as much as possible to minimize the time spent during testing also during regression testing.

A. Automation Feasibility Analysis

This is the first step for automation. The mission software elements feasibility check for automation has to be analysed. The test cases that can be brought under automation has to be segregated. Also, some tests have to carried out manually. Selecting the appropriate test tool is very important Ref [3].

B. Test Script Development

The test scripts have to be generated for the auto testcases. It has to be tested once manually for all possible conditions before automating so that these test cases can be repeated at many phases of the project and also for different projects. The scripts written shall be reusable, and test suite can be generated, well-structured and documented Ref [4]. Unit test cases are to be written first. This is the basic step for automation strategy. Next is the functionality checks and later comes the regression testing and user interface testing.

C. Test Case Execution

In this phase all the test scripts written will be executed and all results will be logged in excel sheet/ text file/ database. The tests are to be executed manually before automating. The status after execution that is pass / fail has to assigned automatically as per the criteria set. If the status is fail then failure analysis has to be carried out.

D. Test Result Generation and Analysis

After the test script execution, the logged test results will be analyzed, the output will be shared with different stakeholders and reports will be generated. The failed test cases have to be clearly analysed, it is the critical part of testing strategy. Well defined process for test result analysis has to be followed to save time and effort. The bugs can be related to test environment, application under test or in the automation script. All the activities of automation strategy should be documented for reference. Figure 2 explains the automation strategy.

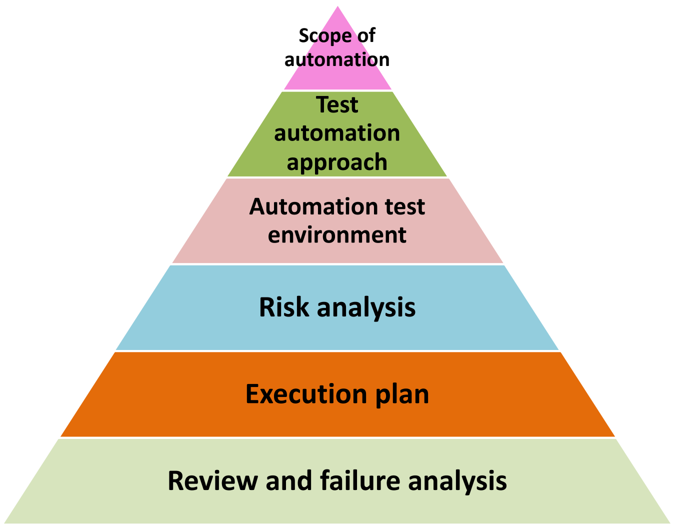

Figure 2: Automation Strategy

E. Risk Analyses

Risk analysis has to be carried out. That is each requirement testing and the risk associated has to be clearly described. The preliminary hazard analysis has to be carried out. Risk level has to be defined later risk priority has to be assigned. Probability of risk occurring has to be estimated. Mitigation technique has to be worked out. Risk analysis has to be worked out at the beginning of the project else the cost is huge if we delay it. Risk list can be dynamic, risks can be added and deleted at any part time in the software development life cycle.

Test automation is the new strategy for faster development cycle. And deliver the bug free software at faster rate with less cost. High quality software will be delivered with this strategy. Test automation life cycle phases has to be incorporated in the testing strategy. Figure 3 shows the test automation life cycle

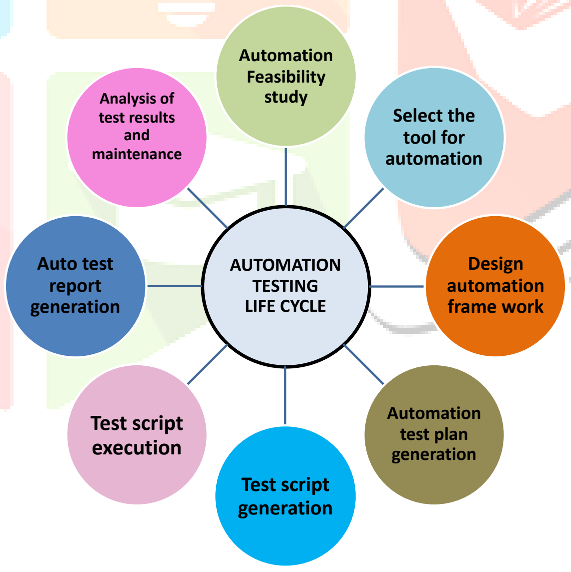
AUTOMATION TESTING LIFE CYCLE
Automation Feasibility study
Select the tool for automation
Design automation frame work
Automation test plan generation
Test script generation
Test script execution
Auto test report generation
Analysis of test results and maintenance

Figure 3: Test automation life cycle

IV. STEPS AND GUIDELINES TO BE FOLLOWED FOR AUTOMATED TESTING

A. Auto test cases and manual test cases has to be categorized

All test cases cannot be automated. The test cases which can be automated are to be identified. These auto test cases are to be clearly written and the input and outputs has to be as per the requirements, because these tests are run more often. Tests may have multiple data sets. That is data collected during initial bench level test, dis assembled, assembled, thermovac, MST are to be tested with the same test case, hence human error in the repetition of test cases can be avoided. Test can be performed fast hence time and cost saving. These tests are run on different hardware and software platforms. This will consume lot of effort and time if we have to do manually Ref [6].

B. Test cases has to be generated from the start of SDLC that is once requirements are finalized

The testing can be started at an earlier stage in the software development life cycle so that more bugs will come out at early stage. As and when the unit testing is completed test suite has to be built up which helps to test again and again, also running many times more bugs can come out.
The test cases that are to be automated are:

- ✓ Test cases which have common functionality across the mission software
- ✓ The complex test cases which will be repeated for many projects
- ✓ Test cases which have large data to be handled
- ✓ Technical feasibility for large amount of code coverage

C. Select the accurate automated testing tool but it is not the solution

There are various automated tools existing. The right tool has to be selected which suits our requirements, it is just the beginning. Selecting a tool does not mean all the test cases can be automated and tested. It does not give the complete solution, only it helps to manage the test cases easily. Tool alone cannot promise the solution for automation; proper skill set is also required. The created automated tests shall be reusable, maintainable and tough to changes in the applications Ref [5].

D. ***Required data for testing has to be collected much in advance***

The data that is used for automated test is stored in external file. It can also be stored in database or in XML files, excel sheet, simple text file. A good automated testing tool shall be capable of handling any of this input files for the test cases. This type of external input files is easy to maintain if input has to be modified/ updated without touching the automated test case. At the same time the output of test cases can also be saved into the excel sheet or text file or into the database. The generated data can be of any data type like Boolean, integer or string. It can be saved to a variable or file. This helps in analyzing the test results which are run at different time, also report generation becomes easier. This data collection helps at later stages if any changes are in requirements, so that the test suit can be modified easily which saves lot of time at the later stage of SDLC.

V. CONCLUSION

The best practice for automation testing implementation is discussed in this strategy for automation. There are various features discussed which helps the best practices. The different types of software testing like functional testing, data driven testing, regression testing, coverage testing, web testing, manual testing gets easier and easier to implement. Automation testing allows to test individual test items also it can be organized in a structure and run in certain order. Once the scripting language is understood it is easy to create run test cases. Some software executes are reused in next project then these test scripts help to run the test cases with new spacecraft data. This is more useful for a mission software quality engineer to re run the test cases. Even if some requirement changes the test scripts can be modified and reused. Lot of project time and cost can be saved with this methodology. It helps a new engineer to understand and learn and re use the test cases in much shorter time. Many bugs can be brought out at earlier stage of software development. Adopting this best practice for automation helps to avoid faults and improves automated testing practice. This helps to test fast and deliver software executes on time.

VI. ACKNOWLEDGMENT

We wish to convey our gratefulness to all our colleagues in Onboard Software Quality Assurance Division, Mission Software Quality Assurance Division and sub system designers for their help, and also desire to thank office staff of Reliability and Quality Assurance of software Group (RQSG) for their support.

REFERENCES

[1] Software engineerinmg, Sommerville, Nineth Edition, 2011

[2] NASA Software Safety Guidebook - NASA-GB-8719.13

[3] Implementing Automated Software Testing: How to Save Time and Lower Costs While Raising Quality by Elfriede Dustin, Thom Garrett, Bernie Gauf

[4] Software Test Automation: Effective Use of Test Execution Tools by Mark Fewster and Dorothy Graham

[5] Software Configuration Management, Addison- Babich, W. A., Wesley, 1986.

[6] Software engineering a practitionars approach, Roger S. Pressman, 7th Edition