To Implement An Impacted Integration Test Monitoring And Control For SLM Software

Sanuja Kulkarni[1], S.T.Patil[2]
[1]Dept of Computer Engineering
[2]Professor, Dept of Computer Engineering
[1, 2] VIT, Pune, India

Abstract- *Developer team has to perform the integration testing on the code they develop. If developer makes some changes in the code and makes the submission of that file it goes on Gerrit repository where the code is reviewed. The code reviewer will review the submitted code. After the review the code gets merged without undergoing integration test. In the existing system takes lot of time in between code submission and running the integration tests. If the integration test fails for any module then its time consuming to identify what is the cause of failure. If some error occur in the system it is very difficult for the developer to find the problem. Developer cannot check what will be the effect of code on the integration tests during the code submission. The failure of the integration test reduces the efficiency of the software. All the functionalities are made available on different module, which reduces the performance of the system. To overcome the above mentioned issue the integration tests should be identified and solved during the time of code submission. Impacted Integration Test Monitoring and Control speed up the testing Process. This tool automates all the existing integration test process. Integrating the Jenkin Job to SLM cases to generate the log reports and Email alert to the developers, scrum masters and testers.*

Keywords- SLM (Software Lifecycle Management), Integration Testing, Jenkins, OpenGrok, Gerrit, GIT.

I. INTRODUCTION

The functioning of Impacted Integration Test Monitoring and Control Tool software Features and can be used by any Developer Team who wants to Create and Automate Integration Test for specific Features of SLM(Software Lifecycle Management). The Impacted Integration Test Monitoring and Control Testing Tool automate integration test cases and Provide Success and Failure Result. This Paper gives the insight into analysis and requirements of the system for determining the operating characteristics of Impacted Integration Test Monitoring and Control Testing Tool. It gives an overview of how the Impacted Integration Test Monitoring and Control Testing Tool software is designed to operate. The intended audience of this Paper is the Developer Team who is going to use Impacted Integration Test Monitoring and Control Tool, The purpose is to Automate Complex and Time Consuming Integration Test Cases.

The purpose of this Paper is to provide details about the system product to the developer. This document gives a detailed description about the system, its features, functionalities, and specifications of the system. It describes different problem statements involved in the system, the product perspective for the intention of the target audience system for which the system is being developed. Different hardware and software specifications that will be required for the system to be built.

For the developers, this Paper will provide the specific requirements. This will help the developer in the development of the system. Product hierarchy will guide the developer to know the exact goals of the system followed by their objectives of each goal. Process given below the objectives will guide the developer to know how to achieve the goals while the development of the system. For the User, the Paper provides product functions. It gives the details of the functionalities of the product. This document would give the overall description of each and every function which the user intent to use.

The purpose of the paper is to provide a detailed overview of our system‘s goals and objectives. It also states the processes required to complete the objectives. This paper list outs different types of user interfaces, different constraints and different assumptions and their dependencies that are important for the implementation of the system. This Paper will make the developer to design the further design the use cases and the sequence diagrams. This paper will be used by both the developer and the software during the software development life cycle period.

1.1 PROJECT SCOPE

Developer should be able to identify the impacted tests and cause of failure to monitor and control the test cases.

It will run the impacted integration test at the time of code submission by automated job using Jenkins. Jenkins job will get triggered on every code submission in the system. It will create Email Alert and Log Results.In the proposed system let the developer know which integration test classes are getting affected by the changed file before the submission.

To reduce the manual overheads and saving the considerable amount of time. All these factors thus increase the future prospects.

To create a framework this will automate integration test process.

To improve the dynamic feature this system this will help to collect all the log files on a server and will help to analyse the report.

To create a process this will track impacted test cases.

To integrate the Jenkins job with SLM to monitor the test failures.

To reduce time consumed in integration testing.

II. LITERATURE SURVEY

To understand the domain of SLM and Integration Testing a literature review was conducted. This chapter aims to explain various approaches for SLM and Integration Testing.

In [1] Author determines an automated approach for generating integration tests from declarative contracts. Our approach is based on the transformation of declarative contracts into an operational model, called function net. It reaps several important benefits that are not reported in the literature. First, the transformation allows correctness analysis of contract specification, which assures soundness of tests to be generated from the contracts. The operational semantics of function nets also better captures the operational nature of test generation process. Function nets are lightweight high-level Petri nets, which are a well-studied formal method with graphical and mathematical notation. Second, the transformation makes it feasible to generate integration test sequences automatically with respect to coverage criteria of contract-based test models.

In [2] Author determines infrastructure and automation-based approach, together with the set of best practices. We then explain implementation details and a new test automation framework architecture that combines several control structures, which is easier to control the workflow of tests and test environments, as well as other reusable components to interface with external systems. We call this framework NTAF, which is an abbreviation for NHN Test Automation Framework. NTAF has gained acceptance in NHN and is now being presented outside NHN. This paper reports on applications of NTAF for supporting automated testing methods in practical products and details the benefits gained from NTAF used in various products.

In [3] Author determines the approach that allows to transform component tests to system tests, supporting the verification of test cases for individual components using only the interface of the overall system. A transformation is not always possible, for a given component test (on component level) an equivalent test case on system level may not exist.

In [4] Author developed an effective framework for automated testing to control the execution of tests. This framework enables implementing of CI practices. It helps automate the distribution, execution, and results analysis of test cases. It aids communication among various stakeholders by using tables to represent tests and test environments. Those who are new to testing do not need to dive right into complex automation tools or test scripts.

In [5] Author determines Integration Testing reflects the attitude of trying to understand the complexity of a software application and wanting to test its individual components as well as the communication of the components with each other. Integration Testing is more than just testing the communications going through the Enterprise Service Bus or any message queues. Literature on testing offers multiple definitions of Integration Testing.

In [6] Author determines that Git is a distributed revision control system available on all mainstream development platforms through a free software license. An important difference of git over its older ancestors is that it elevates the software's revisions to first-class citizens. Developers care deeply about software revisions, and git supports this by giving each developer a complete private copy of the software repository and numerous ways to manage revisions within its context. The ability to associate a local repository with numerous remote ones allows developers and their managers to build a variety of interesting distributed workflows, most of which are impossible to run on a traditional centralized version control system. The local repository also makes git responsive, easy to setup, and able to operate without Internet connectivity. GitHub is a git repository hosting provider that simplifies many repository

management tasks through a Web-based user interface while also promoting cooperation in open source projects.

III. EXISTING SYSTEM

SLM enables access to relevant product information by all parties using permissions to filter content as appropriate and facilitates the delivery of product-centric data throughout the enterprise.

Allowing for easy distribution and update of product, parts and service information, SLM provides in-house service teams or dealer/distributor networks with readily accessible, accurate and timely information that improves service performance.

This SLM product significantly improves the access, usability and accuracy of technical and service parts information.

The existing system is used for integration testing of SLM application

When the developer makes some changes in the code and makes the submission of that file it goes on Gerrit repository where the code is reviewed.

The code reviewer will review the submitted code. After the review the code gets merged without undergoing integration test.

IV. METHODOLOGY

In Existing system when the developer makes some changes in the code and makes the submission of that file it goes on Gerrit repository where the code is reviewed. The code reviewer will review the submitted code. After the review the code gets merged without undergoing integration test. If some error occur in the system it is very difficult for the developer to find the problem

In the current scenario, this Jenkins job does not do regression testing. We are running this build on one virtual machine because of which the job queue is increased. The existing system takes lot of time in between code submission and running the integration tests. In the current scenario if the integration test fails for any module then its time consuming to identify what is the cause of failure. If some error occur in the system it is very difficult for the developer to find the problem. Developer cannot check what will be the effect of code on the integration tests during the code submission.

In SLM change that is been introduced for the current work fashion, so as to increase to overall quality of the software. Due to ease of automation, Jenkins CI tool will be used for automating the process. The proposed system considerably saves time and manual efforts by all means. Similarly the manual overhead of the developers and testers is reduced. Devops engineer can evaluate the Log reports and take timely steps to take corrective actions.

After analyzing the existing system and considering its drawbacks and issues, there was a need to develop a tool which will automate all the existing integration test process. Thus evolved the need of Impacted Integration Test Monitoring and Control. Considering the features and platform independence of java language, this tool would be developed in java.

- Proposed System is an change that is been introduced for the current work fashion, so as to increase to overall quality of the software
- Due to ease of automation, Jenkins CI tool will be used for automating the process.
- The proposed system considerably saves time and manual efforts by all means.
- Similarly the manual overhead of the developers and testers is reduced.
- Devops engineer can evaluate the Log reports and take timely steps to take corrective actions.

Impacted Test Cases Monitoring and Contol

Impacted Test Cases Monitoring and Control:

- Monitoring
- Control

MONITORING:

On error upload and reports server to monitor. On success upload the report with success id on ins-blaine-mirror.

CONTROL:

Tracking the impacted test cases. Proper mail notification is sent to the Developer, corresponding scrum masters, and director regarding the failure of the test case.

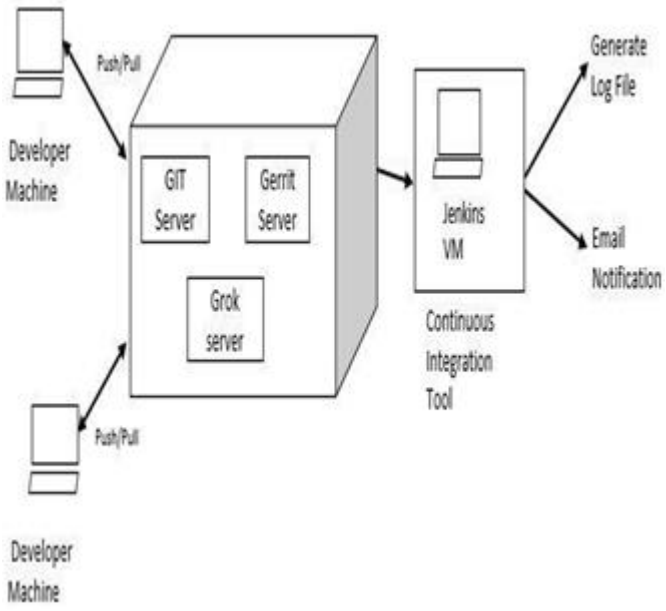

Fig -1: Proposed System

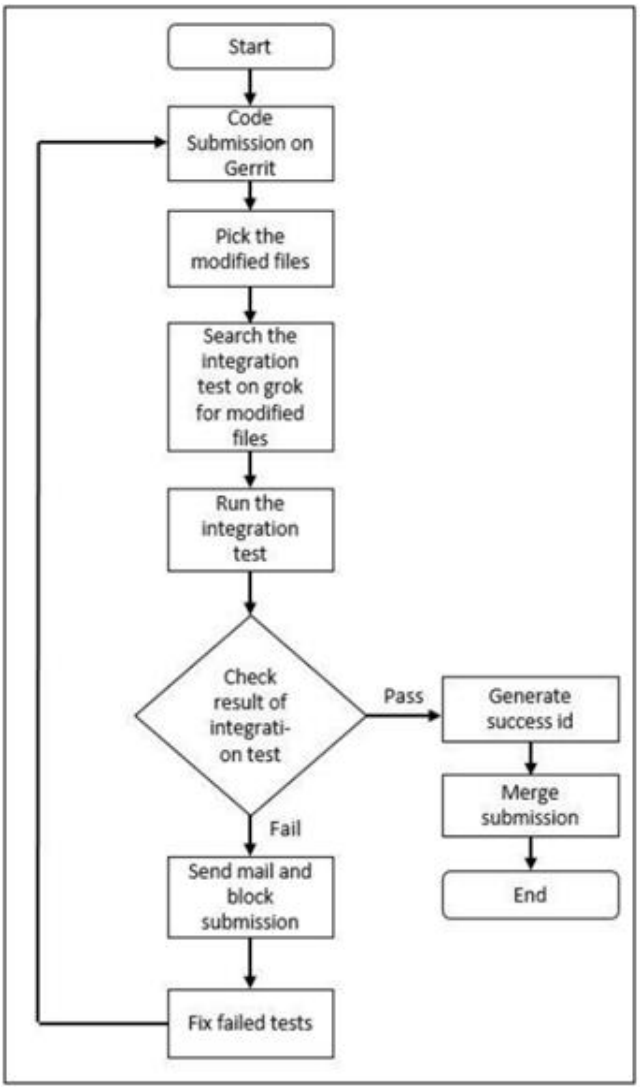

Fig 2: Flow Chart

Test cases for Email:

Notification for valid Test case

Test case ID: Id_integ_2.0				
Test Priority: High		**Test designed Date**:19 April, 2019		
Module name: Email notification				
Test Title: validation of email notification		**Test execution date**: 22 April, 2019		
Description: This test case will validate the email notification after the any of integration test failure.				
Precondition: The integration test cases should fail.				
Step No.	Test Step	Expected Result	Actual Result	Status
1	Fetch modified files			
2	Search integration test for fetch files			
3	Run the integration test files	Test case should fail.	Test case failed.	Pass
4	Email notification	Email notification sent along with submission details	Email notification sent along with submission details	Pass
Post Condition: Email notification will be received by the recipients				

Notification for Invalid Test Case

Test case ID: Id_integ_2.1				
Test Priority: High		**Test designed Date**:19 April, 2019		
Module name: Email notification				
Test Title: validation of email notification		**Test execution date**: 22 April, 2019		
Description: This test case will validate the email notification after the any of integration test failure.				
Precondition: The integration test cases should fail.				
Step No.	Test Step	Expected Result	Actual Result	Status
1	Fetch modified files			
2	Search integration test for fetch files			
3	Run the integration test files	Test case should fail	Test case failed.	Pass
4	Email notification	Email notification sent along with submission details	Email notification not sent along with submission details	Fail
Post Condition: Email notification will not be received by the recipients				

V. RESULTS

```
Directory:Core
TimeStamp:Mon Apr 22 04:01:58 PM IST 2019 Owner:Sanuja Kulkarni
Sub_Id:24215
Patch_Id:1
BUILD SUCCESSFUL
Success_Id:12076
Directory:WebServices
TimeStamp:Mon Apr 22 04:06:47 PM IST 2019 Owner:Sanuja Kulkarni
Sub_Id:24215
Patch_Id:1
BUILD SUCCESSFUL
Success_Id:12077
```

Fig 3: Test Passed and Log file with success id

```
Directory:Core
TimeStamp:Mon Apr 22 04:10:54 PM IST 2019 Owner: Sanuja Kulkarni
Sub_Id:24215
Patch_Id:1
BUILD FAILED
Directory:WebServices
TimeStamp:Mon Apr 22 04:06:47 PM IST 2019 Owner: Sanuja Kulkarni
Sub_Id:24215
Patch_Id:1
BUILD SUCCESSFUL
Success_Id:12077
```

Fig 4: Test failed and Log file without success id

VI. CONCLUSION

Impacted Integration Tests should be able to monitor and track the integration test cases and reduce the manual overheads and saving the considerable amount of time. And improves the dynamic feature this system, and helps to collect all the log reports on a server to analyse it. It notifies the developer know which integration test classes are getting affected by the changed file before the submission.

REFERENCES

[1] Dianxiang Xu, Weifeng Xu, Manghui Tu "Automated Generation of Integration Test Sequences from Logical Contracts, IEEE 38th Annual International Computers, Software and Applications Conference Workshops

[2] Eun Ha Kim, Jong Chae Na, and Seok Moon Ryoo, "Implementing an Effective Test Automation Framework", 33rd Annual IEEE International Computer Software and Applications Conference

[3] Bernhard Sch¨atz Christian Pfaller, "Integrating Component Tests to System Tests", Electronic Notes in Theoretical Computer Science 260 225–241

[4] Eun Ha Kim, Jong Chae Na, Seok Moon Ryoo, "Test Automation Framework for Implementing Continuous Integration", Sixth International Conference on Information Technology: New Generations

[5] The Testing Consultancy," Integration Testing White Paper", TTC

[6] Diomidis Spinellis, "GIT", 34th Annual IEEE International Computer Software and Applications Conference