ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)
Vol. 4, Issue 2 (Apr. - Jun. 2016)
International Journal of Advanced Research in Computer Science & Technology (IJARCST 2016)

Parallel Computing for Continuous Regression Automation Framework

[I]Omkar P, [II]Rampur Srinath
[I]PG Student, [II]Associate Professor,
[I,II]Dept. of Information Science & Engineering, The National Institute of Engg., Mysuru, India

Abstract
Regression Automated testing framework is one of the key fields of testing which focuses on high testability to achieve desirable goal. The primary focus of this paper is to advocate the use of end-to end based technology for the Regression Testing using CR framework and to enable the parallel execution of test to enhance the productivity and reduce the time consumption of testing. This paper presents Jenkins based continues integration test framework for enhancing the efficiency of Regression Testing for Linux distribution builds. Different methodologies has been proposed for Design Comparison, Notification of build availability, Code Comparison, Impact analysis, Parallel execution approach, Test Case Generation and gathering of result of testing. The results can be viewed by using JBAF platform.

Keywords
Parallel Computing; Continuous Regression, Jenkins Based Automation Framework, Cartesianer, Augmenter,SUT

I. Introduction

Software development life cycle is a process based approach of determining the expected behavior of the program. Software testing is the key part in the life cycle of any software development [1]. To provide the establishment of test criteria and to evaluate the performance of it to determine whether the desired criteria of test have been met or not and to create the response based on the test. High accurate testability is necessity goal of the life cycle of the product [5].

Most of the testing tools that are available do not support dynamism and independence. So, there is a need to develop a framework which would support the dynamic behavior, minimizes the error and provides better efficiency. In continuous integration environment with regression testing is bounded by tight time constraints [2]. To satisfy time constraints and achieve goal of testing, build test cases must be efficiently arranged in execution. Prioritization techniques are commonly used to arrange test cases to reflect its importance according to one or more criteria. Reduced time to test or high fault detection rate are such important criteria.

Linux is facing a quandary with exponentially growing Hardware & Software coalescences, Expanding Hardware platforms and exploding amalgamations of Hardware, Firmware and Software. In an ever changing IT it's important to test and report about the software product among all the platforms. Performance of the computation is improved by approaching towards applying parallel computing in which the tasks run in parallel on multiple nodes [3]. Parallel computing displays significant potential to distribute cost-efficacious solutions for applications which demands high performance [4]. However, performance often falls well between utilizer prospects and below what would be considered cost-efficacious. This paper present a framework approach (Parallel computing for Continuous Regression Testing framework) to enhance the efficiency of continuous regression testing.

II. Problem Statement

The Linux operating system (OS) is one of the leading OS in the world which is available to the end user in enterprise and open source edition. The popularity and accuracy of the Linux lead to its existence in most of the servers all over the world and IT industries are using one or another Linux distribution for their businesses. To provide the best accurate system performance, each and individual components of the Linux distribution is well tested prior to the release. Such software testing tasks, with much extensive low level interface uses regression testing, which is time consuming and laborious to carry out manually. As mentioned above due to ever-changing IT world, it's very difficult to look for required updates frequently, test and validate on available platforms and hardware form manual test environment. Manual testing for such a laborious flow of testing which requires much effort. A test that requires significantly more resources is a regression test, which consists of building all easy configuration files distributed with Easy Build, in a pristine install directory. To run a full regression test which is time consuming with single process execution.

III. Proposed Solution

To provide an optimal solution for the above stated problem considering optimal time for testing with cost effective approach through the automation framework. The process of automation is defined in way that all the manual test flow is automated. An Integrated system with well-defined set of rules of automation of specific product, which is the heart of the software testing environment that specifies the way in which the product will be tested is known to be automation framework[1,2].

In the flow of the framework which uses the special testing tools for test automation which helps automated task to control test execution and to compare with the predicated result with the actual result of execution and submits the outcome of the compression back to the framework. Parallel computing displays significant potential to distribute cost-efficacious solutions for applications which demands high performance (as shown in Fig 1) [4]. However, current scenario shows that performance often falls well between utilizer prospects and below what would be considered cost-efficacious. As a result of, Massively Parallel Computer (MPC) hardware and software designers need to concentrate on the factors which affects parallel computing efficiency.

ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)

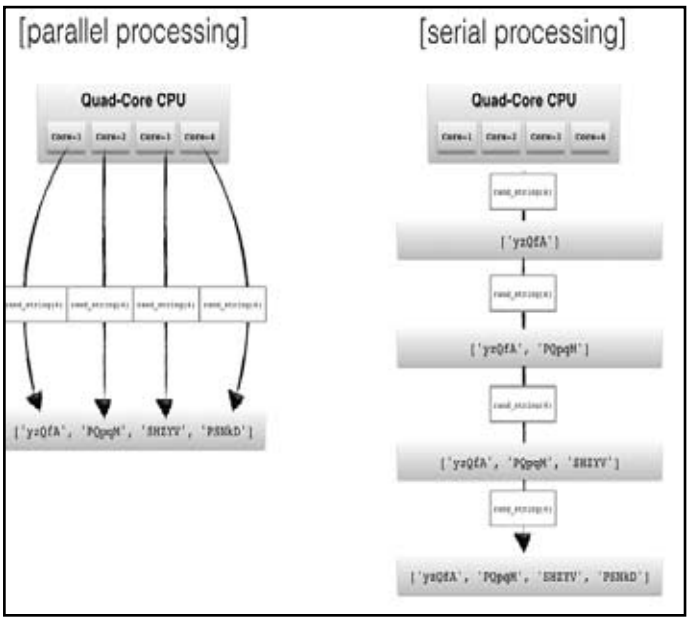

Fig. 1: Parallel and Serial processing workflow.

To provide parallel executions of the system software testing on the system under test using testing a framework is a challenging task, given the framework's sizably voluminous variability. To overcome these challenges and difficulties an automated regression test framework is being developed with a simple Goal of End-to-End automated perpetual parallel computing of regression test framework to facilitate optimal testing of multiple flavors of Linux under different Firmware levels & Operating environments.

IV. Methodology

To address the concerns related to the design of parallel computing regression test of Linux distribution builds a well-designed framework with better testing, maintaining and reporting tools can be approached [6]. This paper presents Jenkins based framework due to its functionality and plugins support. Jenkins is open-source continuous integration software tool testing and reporting tool on isolated changes in a larger code base in real time.

Jenkins plugins have been released for Jenkins framework that extend its use to projects written in several programming languages. Plugins are available for integrating Jenkins with most version of control systems and big databases. Many build tools are supported via their respective plugins. Plugins changes the way Jenkins looks or add new functionality [9]. Builds can generate test reports in various formats supported by plugins and Jenkins can display the reports and generate trends and render them in the GUI.

The software enables developers to find and solve defects in a code-base rapidly and to automate testing of their builds [7]. To build continuous regression framework on top of Continuous Integration, a Continuous Deployment/Delivery is conducted and after a successful test run, it instantly and automatically releases the latest version of the codebase that makes deployment a non-issue and helps to speed up the development.

V. Framework Architecture

The design methodology of parallel regression testing tool consisting of intelligent autonomous flow, which would not only reduce human effort of software maintenance but at the same time it would perform desired impact analysis and increase the efficiency of the system. As mentioned above the Jenkins will support such kind of framework. An architecture framework organizes higher-level structures of the proposed end-to end automation framework are discussed below (as shown in Fig 2).

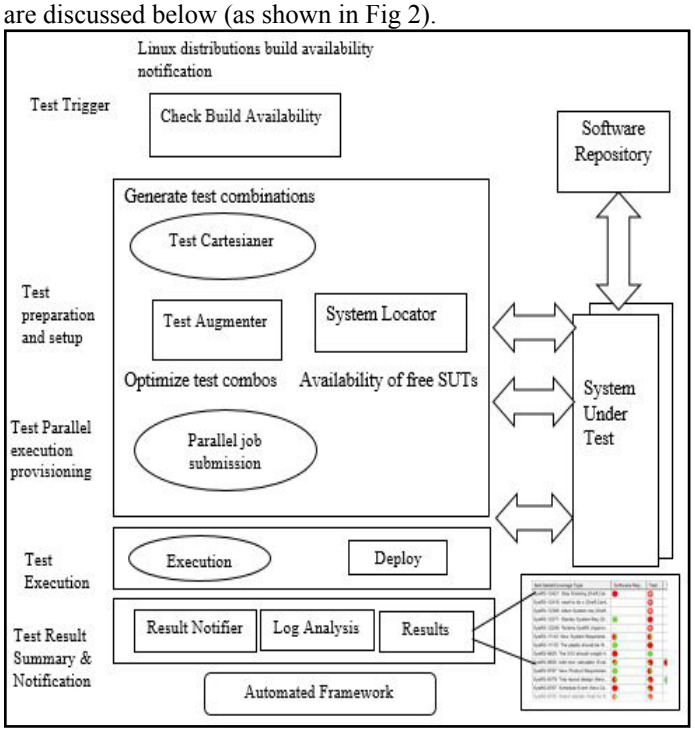

Fig 2: Parallel Computing for Continuous Regression Automaton Framework

For designing and implementing this software testing Jenkins automated framework, methodology adopted includes the following steps:

Test Trigger Phase

The build notifier looks for the availability of the builds for the Linux distribution and notifies to the framework about the build details and download the notified build and present it to the test preparation phase.

Test Preparation and Setup Phase:

This phase has several mutual sub phases which work independently, but support one another in various sections of the framework. Test Cartesianer will generate test combinations pertaining to available build which includes necessary test buckets pertaining to the build with relevant control and source files and pass on to the Test Augmenter.

The Test Augmenter focus on structuring test arguments with which test buckets should run on the build, the type of system on which they need to be executed and most important is that it optimizes the test combinations which are driven from the test cartesianer. The test augmenter will make optimized test combinations as test arguments jobs and stores it into the reference list so that will be refereed for executions which is likely an shell command with the argument passed.

System locator which refers the list constructed by the augmenter depending on the need of the build, it locates the availability of free system under test (SUT) systems in the resource pool which test argument demands. The reference is made of available SUT with its resource capacity.

Test Parallel Execution Provisioning Phase:

This phase is key to provide the better performance which takes both references from test augmenter and system locator. This phase links the test argument jobs in the reference list to its appropriate systems with reference from the system locator created by the parallel job submission by using multiprocessing to facilitate the faster execution to display significant potential to distribute cost-efficacious solutions. Test parallel execution provisioning will provide the parallel job submission in accordance with the system resource constraint and build requirements.

Test Execution Phase

Parallel job submission with SUT reference is provided to the execution phase. The test execution creates required platform such as creation of node on the system and deploy the necessary Linux distribution and software from the software repository before the execution of the regression test on the build for the selected environment. The execution of the test for the build will begin once the deployment of necessary environment is done on the system.

Test Result Summary & Notification Phase

The result of the test is analyzed through the log analysis of the test and through Jenkins reporting user interface. Jenkins is also well known for the maintaining and reporting tool which makes reporting of all the test builds and maintains the order in which it can be reported back to the developer (as shown in Figure 3). Result Notifier acts as the plugin for the Jenkins framework which notifies the detailed report of the test results of build back to the developer, which helps to maintain the lifecycle of the build efficiently.

Fig 3: Jenkins UI for result analysis

The flowchart of the Automaton Framework (as shown in Fig 4). The Framework is end-to end automated such that build availability triggers the regression test automatically to move to the test preparation and setup phase such as test cartesianer and augmenter then execute the parallel job submission on the located SUT. From the software repository, required software are deployed then build test executed and result is reported. The framework will support for verification, validation, and productive capability of build [8].

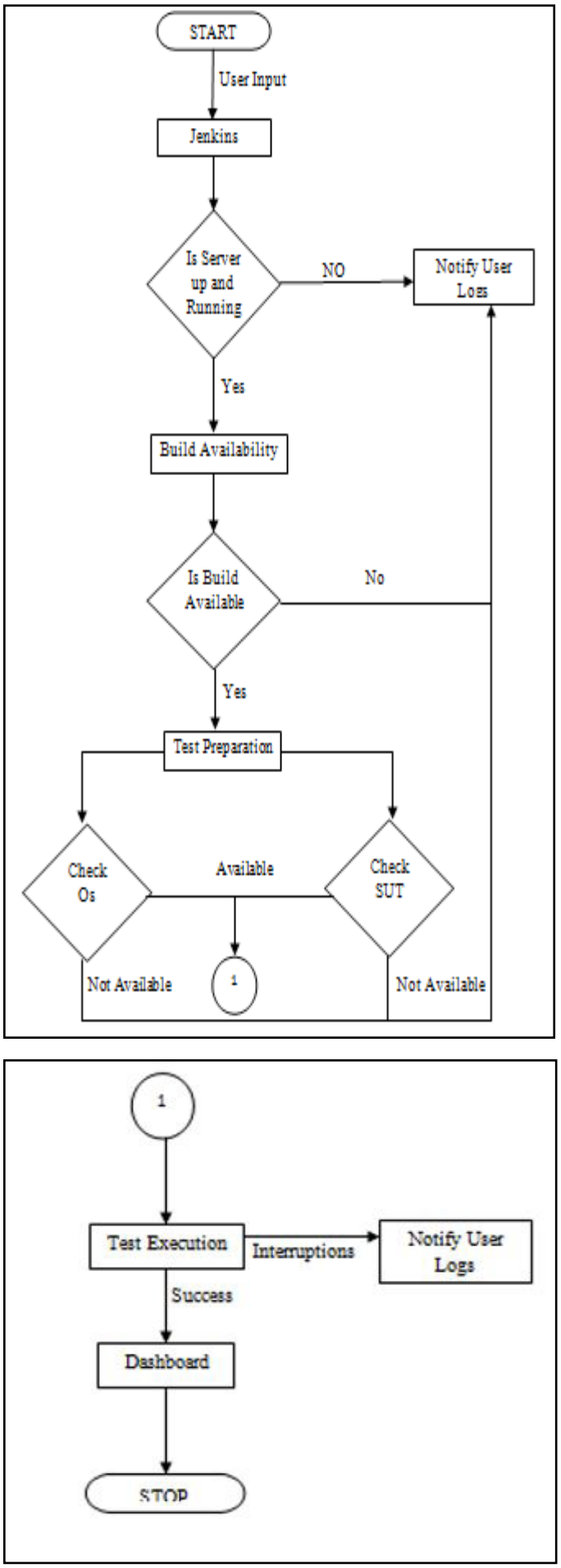

Fig. 4: Flowchart of the Automaton Framework

ISSN : 2347 - 8446 (Online)
ISSN : 2347 - 9817 (Print)

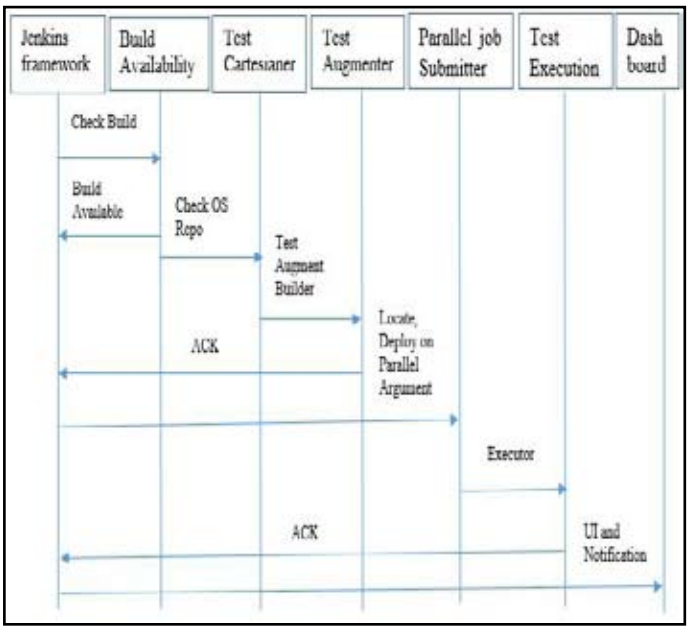

Fig 5: Sequence diagram of Framework.

The sequence diagram of Jerkins framework (as shown in Fig 5), it arbitrarily checks for the availability of build using Build Availability notifier. When there is availability of build then it sends build notification to framework. Once the build is available then it checks for the corresponding linux distribution OS in the software repository. After it checks for the OS then test augment builder is carried out at Test Augmenter in corresponding with the availability of the system for the testing and an acknowledgement is sent to the Jenkins Framework saying that test preparation and setup is carried out. Then Parallel job Submitter locates systems and deploys it on the parallel augmenter and then its executed. After execution an acknowledgement is sent to the Jenkins Framework about the detailed report of the execution. At last, the evaluation of the automated test execution is well reported by Jenkins and error handling is well taken care of. Jenkins will provides evaluated result of test in its Dashboard as user interface and notification is sent regarding evaluation of the build to the corresponding developer with detailed report.

VI. Performance Evaluation

Fig 6: Generic dashboard for evaluation test execution.

A Jenkins generic dashboard to display the results (as shown in Fig 6) of build test execution will enlighten the developer and makes for easy understanding of the performance evaluation of the build and bug fixing.

VII. Conclusion

This paper presents parallel computing regression testing framework for enhancing the efficiency of Testing so as to minimize the errors and to provide the well documented report of build test execution. The JBAF has control over all phases of the test framework and co-ordinates with all sub-phase such that it provides the accurate result analysis of the build. The outcome of the framework gives the flexibility to the user to configure according to their needs and at the same time optimizing resources with improved performance.

References

[1] *Wikipedia, "Test Automation Framework - Wikipedia, the free encyclopedia", 2008, Available: https://en.wikipedia.org/wiki/Test_automation#Framework_approach_in_automation*

[2] *A. K. Onoma, W. T. Tsai M. Poonawala and H. Suganuma,"Regression Testing in an Industrial Environment", Communications of the ACM*

[3] *Martin Fowler, "Refactoring: Improving the Design of Existing Code", Second Edition, pp: 71-18.*

[4] *G.M. Amdahl, "Validity of the Single Processor Approach to Achieving Large Scale Computing Capabilities", Re. fr. the AFIPS Conf. Proc., Vol. 30 (Atlantic City, N.J.), AFIPS Press, Reston, Va., vol. 12, no. 3, pp. 483-485*

[5] *R. Baxter, "Software Engineering Is Software Engineering," Proc. 1st Int'l Workshop Software Eng. For High Performance Computing System Applications, IEEE CS, 2004, pp. 14–18.*

[6] *T. Xie and D. Notkin, "Checking Inside the Black Box: Regression Testing by Comparing Value Spectra," IEEE Trans. Software Eng., vol. 31, no. 10, 2005, pp. 869–883.*

[7] *H. Remmel et al., "Supporting the Testing of Scientific Frameworks with Software Product Line Engineering: A Proposed Approach," Proc. 4th Int'l Workshop Software Eng. for Computational Science and Eng., ACM Press, 2011, pp. 10–18.*

[8] *W.L. Oberkampf et al., "Verification, Validation, and Predictive Capability in Computational Engineering and Physics," Applied Mechanics Rev., vol. 57, no. 5, 2004, pp. 345–384.*

[9] *Jenkins to automate for Execute repetitive tasks, save time, and optimize your development process, Available: https://github.com/jenkinsci/*