A Comprehensive Survey on Continuous Regression Testing Tool

Rakshith KN[#1], Dr. T H Sreenivas[#2]
#1 PG student, Department of IS&E,
The National Institute of Engineering, Mysore, India
#2 Professor, Department of IS&E,
The National Institute of Engineering, Mysore, India

Abstract— Regression testing is the testing in which the existing software applications is used to make sure that a change or addition hasn't broken any existing functionality. Which will download the required Linux Distro into the specified path and check the coordination with the machine. The different testing techniques, continuous regression testing tool methods and its advantages are discussed in this paper.

***Index Terms* — Regression testing, Types of Testing, Distro**

I. INTRODUCTION

Linux on Power test teams within Linux Technology Center are facing a problem with exponentially growing Hardware & Software combinations, Expanding Hardware platforms, and Exploding combinations of Hardware, Firmware and Software. In addition the test teams are required to frequently validate several generally available Linux distributions on newer platforms and hardware. To overcome these challenges and difficulties an automated regression test framework is being developed with a simple Goal - End to End automated continuous regression test framework. The Project is titled Continuous Regression Test Framework as we continuously check for the Various Releases of Linux Which Can be either LTS or NON LTS, Once we have these releases we download both the Iso images and NetBoot files which are later installed on the Hardware in following three categories namely Bare Metal ,POWER VM and POWER KVM.

II. BACKGROUND

Regression means retesting the unchanged parts of the application. Test cases are re-executed in order to check whether previous functionality of application is working fine and new changes have not introduced any new bugs. This test can be performed on a new build when there is significant change in original functionality or even a single bug fix. This is the method of verification. Verifying that the bugs are fixed and the newly added features have not created in problem in previous working version of software. [4]

Earlier version was quite tedious mainly because it consisted to download each and every version of Linux available in the market and test the each and every module individually, as a result of which it took so much time to test on every machine every possible combination of test cases.

III. EXISTING TESTING TECHNIQUES

1. Random Testing

Is a black-box software testing technique where programs are tested by generating random, independent inputs. Results of the output are compared against software specifications to verify that the test output is pass or fail. In case of absence of specifications the exceptions of the language are used which means if an exception arises during test execution then it means there is a fault in the program. [1]

2. Functional Testing

Is a quality assurance (QA) process and a type of black-box testing that bases its test cases on the specifications of the software component under test. Functions are tested by feeding them input and examining the output, and internal program structure is rarely considered (not like in white-box testing). Functional testing usually describes *what* the system does. [1]

3. Control flow Testing

Is a structural testing strategy that uses the program's control flow as a model. Control-flow testing techniques are based on judiciously selecting a set of test paths through the program. The set of paths chosen is used to achieve a certain measure of testing thoroughness. [2]

4. Data flow Testing

Is a family of test strategies based on selecting paths through the program's control flow in order to explore sequences of events related to the status of variables or data objects. Dataflow Testing focuses on the points at which variables receive values and the points at which these values are used. [3]

5. Mutation Testing

This Mutation Testing is a fault – based testing technique which is used for testing software at the unit level, integration level and specification level. Mutation Testing provides a testing criterion called the mutation adequacy score. This score can be used to measure the ability to detect faults. This process is called mutation analysis. In mutation testing, the faults represent the mistakes made by a programmer so they are deliberately introduced set of faulty programs called mutants. [4]

6. End-to-end Testing

End-to-end testing, tests a completely integrated system to verify that the system meets its requirements. For example, a system test might involve testing a logon interface, then creating and editing an entry, plus sending or printing results, followed by summary processing or deletion (or archiving) of entries, then logoff. [1]

7. Stress Testing

Stress testing is a form of deliberately intense or thorough testing used to determine the stability of a given system or entity. It involves testing beyond normal operational capacity, often to a breaking point, in order to observe the results. [1]

8. Sanity Testing

Sanity test or sanity check is a basic test to quickly evaluate whether a claim or the result of a calculation can possibly be true. It is a simple check to see if the produced material is rational. The point of a sanity test is to rule out certain classes of obviously false results, not to catch every possible error. A may be checked to perform the test. The advantage of a sanity test, over performing a complete or rigorous test, is speed. [1]

9. Acceptance Testing

Acceptance testing is a test conducted to determine if the requirements of a specification or contract are met.[1]

10. Usability Testing

Performance testing is generally executed to determine how a system or sub-system performs in terms of responsiveness and stability under a particular workload. It can also serve to investigate, measure, validate or verify other quality attributes of the system, such as scalability, reliability and resource usage. [1]

IV. Steps involved

The whole process mainly consists of three fundamental steps that will be carried in the method.

1. Build Availability : The purpose of this module is to check for the Availability of a Particular Linux Release. The Linux Release can be LTS release or the Non LTS which are Long Term Support and Non Long Term Support respectively. Once the Build is available the Particular release will be downloaded and placed in a repository whose download path will be specified in the code. It checks for the updates in the release for every week.

2. Test Preparation : The purpose of this module is to check for the various availability of the system for OS installation and generate optimal test combinations to be executed on the system , which will be used to test for the many combinations.

3. Test Execution : The main purpose of this is to deploy necessary software and start executing the required tests. In this phase all the combination of hardware and software is tested and checked if all the combinations are workign as per required. If not it will provide the required information as which all combinations are not working.

V.Procedure Of CRTL

Procedure of Continuous Regression Testing Tool

1. Once the tool is developed it will be continuously running without any interventions.

2. Once the time is set in the code then the script will call by itself as a result of which no manual interaction is required.

3. Script will automatically download the specified distro into the repository.

4. It will check every once in a while based on the requirement or if its a LTS or Non – LTS, and if the latest version is already present, then it will not download the distro.

5. As a result of ehich it will download the iso file and netboot file into the repository given in the specified path.

6. Once downloaded it will be installed into the required machine.

7. Once installation procedure is completed it will check all the requirements and support of the system.

8. If there is any problem in the machine with that particular distro it will provide the updated information regarding that.

VI. Conclusion

Regression Testing plays a major role in the whole procedure. It is well suited for the CRTL procedure in which various combination of hardware and software are tested in the framework, which considerable reduces the execution of the test cases which was executed in parallel there by having an upper hand on serial execution which takes considerably more time for execution. Due to the automation it reduces less execution time and man power which in turn will reduce the overall maintenance cost of the complete procedure. It is also made sure that each and every output of a particular instruction that is being executed is proper logged on to user logs for further analysis and feedbacks. The Test Framework can also be developed on other platforms such as windows and iOS, to aid optimal testing of various combinations of hardware and software.

ACKNOWLEDGMENT

We are greatly thankful to our family and friends for the continuous support that has provided a healthy environment to drive us to do this project. We also thank the Principal and the Management of NIE for extending support to this work.

REFERENCES

[1] Wikipedia - https://en.wikipedia.org/
[2] Topics in Control-Flow Testing - Material drawn from [Beizer, Mancoridis]
[3]www.tutorialspoint.com/software_testing_dictionary/data_flow_testing.htm
[4] A Review of Software Testing Techniques – Manpreet Kaur and Rupinder Singh
[5]http://www.softwaretestinghelp.com/regression-testing-tools-and-methods/

AUTHOR PROFILE

Rakshith KN was born in Mysore,India (1990). He obtained B.E. in 2012. He is presently a student of M.Tech at National Institute of Engineering , Mysore in Computer Network Engineering. His research interests are in the field of Server Machines.

Dr.T.H.Sreenivas is Professor in the Department of Information Science & Engineering. He has received his Ph.D. from IIT, Madras, M.Tech. from IIT, Kanpur and B.E. from University of Mysore.
His teaching and research interests are in the field of Operating Systems, Networking, Schedule Optimization and Wireless Sensor Network.