Research article **Available online www.ijsrr.org** **ISSN: 2279–0543**

International Journal of Scientific Research and Reviews

An Exploration from Manual Testing to Automation Testing in Software Industry

Mandara Nagendra[1*], C N Chinnaswamy[2] and T H Sreenivas[3]

[1] M. Tech student, Department of Information Science and Engineering, The National Institute of Engineering, Mysore, Karnataka, India
[2]Associate Professor, Department of Computer Science and Engineering, VVCE, Mysore, Karnataka, India

ABSTRACT

Framework level test mechanization has experienced different ages, where each new age has raised the dimension of deliberation utilized in the test structure. As programming is refreshed or changed, or reused on an altered target, rise of new blames as well as re-development of old issues is very normal. Regression testing is an indispensable piece of the extraordinary programming advancement strategy. Testing is a vital piece of any fruitful programming venture. The sort of testing (manual or automated) relies upon different variables, including venture necessities, spending plan, course of events, ability, and reasonableness. Three imperative variables of any venture are obviously time, cost, and quality - the objective of any effective task is to decrease the expense and time required to finish it effectively while keeping up quality yield. With regards to testing, one sort may achieve this objective superior to the next.

KEYWORDS: Framework, Regression testing, Manual testing, Automation testing.

***Corresponding Author**

Ms. Mandara Nagendra

M.Tech student, Department of Information Science & Engineering,

The National Institute of Engineering,

Mysore – 570008, Karnataka, INDIA

Email: mandara.nagendra10@gmail.com, Mob No. - 8762206936

INTRODUCTION

To incorporate a part inside a bigger framework, three noteworthy properties, the wellness, the accuracy, and the strength, must be tried. The wellness of a segment for an application is when all is said in done treated as the similarity of the gave interface of the part and the detail of the required interface of the application. The rightness of a segment is its capacity to restore the right yield when furnished with the right information, while the vigor concerns the nonappearance of a conduct perhaps endangering whatever is left of the framework, particularly under wrong information. Exactly when bundle of portions is accessible, coordination testing ended up being entirely eccentric and one of the item progression improvement steps identifies with testing process overhauls which should scarcely be conceivable without test automation.

Framework level test mechanization has experienced numerous ages, where each new age has raised the dimension of deliberation utilized in the test plan. The cutting edge test computerization, the watchword driven testing process, abstracts the execution of tests behind abnormal state activities, for example watchwords. In GUI testing, watchwords commonly delineate essential client activities, for example, squeezing keys, composing or perusing content. The tests are worked as successions of watchwords, and catchphrases are consequently converted into solid low-level contents.

There are different devices for test computerization accessible – business and open source, yet few are reasonable for discovery testing. A considerable lot of accessible devices are most appropriate for the unit tests performed by the engineers. With regards to the joining testing or useful check – not all that numerous apparatuses are accessible.

The Robot framework is an open source test automation framework, solely developed by Nokia Siemens communication technology limited company. Robot framework is used for acceptance testing and acceptance test-driven development (ATDD). It has a tabular test data syntax and it uses the keyword-driven testing approach. [10]

Market designs with solicitations for snappier time-to-promote and higher quality programming continue exhibiting troubles for programming associations that consistently work with manual test practices that can't remain mindful of extending market demands. Associations are furthermore tried by their own systems that are normally Graphical UI (GUI) heightened and as needs be staggering and expensive to test, especially since writing computer programs is slanted to advancing necessities, support, refactoring, etc., which requires wide backslide testing.[1]

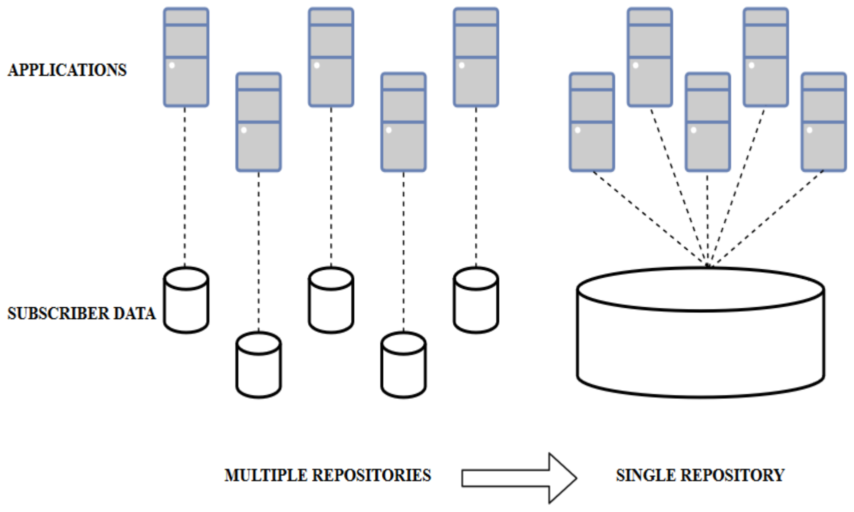

Fig 1. One-NDS in real world

The subscriber information speaks to the data about the overhauled elements of a correspondence service provider, including the subscriber's system distinguishing proof and reference, related instalment model, offset and status with the applications to which they bought in. Figure 1 shows the significance of One-NDS.

One-NDS merges urgent supporter profile data all through systems and contributions to pick up efficiencies and value funds by means of separating realities, storehouses and bringing down the scope of exclusive interfaces among bundles and the measurements they require.

LITERATURE SURVEY

The composition outline in any endeavour structures the explanation behind perception of the endeavour. Thus, before starting any endeavour, far reaching proportion of time is placed assets into understanding the necessities of undertaking, current market for endeavour, unmistakable elective courses of action exist for given issue, etc. This fragment will cover few of the papers that were helpful in my proposition work.

Software testing[2] alludes to the utilization of manual or programmed intends to run a test framework or procedure. Its motivation is to test whether it meets the predetermined prerequisites or discover the contrast between the expectation results and the real outcomes.

Software testing begins with distinguishing proof of test situations from necessities and use cases and making of manual experiments from those situations.

What automated testing[3] offers over that is the chronicle of repeatable and reliable tests over a scope of client bolstered innovation. When the underlying undertaking of making a mechanized test was finished, the computerized test ought to be reusable, repeatable and autonomous of different tests, to be kept running as a feature of a test suite or exclusively. Having tests directed by a machine with minimal human mediation is engaging for a few reasons. Likewise, computerized tests are reliable. Human analyzers can be in some cases conflicting.

The difficulties incorporate upkeep of test contents. An adjustment in interface, new contents should be created and existing must be returned to. Except if the contents are reused, it would be a negligible waste concerning the underlying venture of time recorded as a hard copy the contents.[11]

The Robot framework[4] is an open source test robotization structure, exclusively created by Nokia Siemens. correspondence innovation constrained organization. Robot structure is utilized for acknowledgment testing and acknowledgment test-driven advancement (ATDD). It has a forbidden test information linguistic structure and it utilizes the watchword driven testing approach.[5]

Keyword driven testing or some call it table-driven testing are the ideas generally connected to an application-free mechanization. The analyzer needs to create information tables with catchphrases, autonomous of the test automation framework or some other instrument used to run them. At that point it is required to code the test content that will, in its turn "drive" the tried application and the information.

Selenium2Library is another additional preferred standpoint. Selenium2Library is a web testing library for Robot Framework that utilizes the Selenium apparatus inside. Finding components are made less demanding. All catchphrases in Selenium2Library that need to interface with a component on a site page take a contention ordinarily named locator that indicates how to discover the component. Frequently the locator is given as a string utilizing the locator language structure depicted beneath yet utilizing Web Elements is conceivable as well.

Every one of the highlights, referenced underneath guarantee that Robot Framework can be utilized to mechanize experiments in a snappy and capable manner: High-Level Architecture, Simple Tabular Syntax, Data-driven Test Cases, Separate Test Data Editor, Clear Reports, Detailed logs, Generic test libraries, Web testing, Swing, SWT, Windows GUIs, databases, SSH, Telnet, Remote test libraries and different modules for Jenkins/Hudson, Maven, Ant, Text supervisor support: Emacs, Vim, Text Mate.

With all these, Robot Framework is by all accounts simple to run device for mechanization with such a large number of geniuses and not all that numerous cons. This is an aid to computerization. Any item on overhauls or any updates, experiments should be returned to, in such cases this is beneficial and efficient.

SYSTEM ANALYSIS

Market designs with solicitations for speedier time-to-publicize and higher quality programming continue introducing challenges for programming associations that much of the time work with manual test practices that can't remain mindful of growing business sector demands. Associations are also tried by their own one of a kind structures that are much of the time Graphical UI (GUI) heightened and along these lines erratic and expensive to test, especially since writing computer programs is slanted to developing essentials, support, refactoring, etc., which requires expansive regression testing.[6]

The existing system is chiefly with manual testing, which is tedious and requires progressively human exertion. Testing in the main stage focused on creating unit tests and manual exploratory testing at the GUI level. Unit tests were made for non-UI code and filled in as relapse tests in constant consistently gathers. They were the foundation of our test computerization. Regardless, most features of the virtual support depicted in customer stories must be attempted at GUI level.

Testing is traditionally considered a process that relies upon executing test cases that are carefully designed using test case design techniques.[7,8,9]

Tests are automated to speedup execution cycles, give brief criticism, free testers from tedious errands and diminish human exertion. The primary objective is to automate the relentless undertaking of running the tests. Characterizing tests and executing them as test contents stay manual assignments.

By and large, framework confirmation tests itself takes 5months. There are various One-NDS discharge every year and one noteworthy One-NDS discharge. Framework check tests incorporate execution tests, security tests, robustness tests, load tests, relapse tests, and so forth. There are around 2500 experiments which would take 5 months to execute physically. At present, 250-300 experiments are robotized. The tests include regression testing, load testing, repeated execution and performance testing. These tests are time consuming to be performed manually.

Regression Testing: Here, mechanized testing is reasonable as a result of continuous code changes and the capacity to run the relapses in an opportune way.

Load Testing: Automated testing is likewise the most ideal approach to finish the testing effectively with regards to stack testing. Get familiar with burden testing with our prescribed procedures control.

Repeated Execution: Testing which requires the repeated execution of an undertaking is best mechanized.

Performance Testing: Similarly, testing which requires the reproduction of thousands of simultaneous clients requires mechanization.

In the proposed framework, to diminish the discharge time and to perform dreary manual undertakings through computerization is wanted to accomplish. With the computerization level going up to half, the discharge time can be diminished to 3months.

We made test automation one walk further and associated test age to a GUI-based application created in an extensive industry adventure. Various affiliations have rolled out the improvement from manual to mechanized testing. Test mechanization is an obvious prerequisite in deft improvement conditions. Tests are robotized to speedup execution cycles, give brief analysis, free analyzers from repetitive assignments and decrease human effort. The rule objective is to computerize the steady endeavour of running the tests. Portraying tests and executing them as test substance remain manual assignments.

SYSTEM DESIGN

The general structure of the plan procedure is delineated in the stream graph Fig 2 beneath. The framework configuration has the accompanying parts:

- Robot IDE
- One-NDS labs
- Simulator device
- Latency Simulator
- Test wins

Administrator - it regulates setting up an inventory and the components of all of the One-NDS sections.

Back End database (DB) – the database holds the endorser data for all administrations. Each BE-DB addresses a reasonable improvement that stores endorser profile data, that is, a piece of the hard and fast number of the served supporters.

Routing database (DB) - the Routing DB stores get to keys and references to database segments (supporter profile data).

Provisioning delineates the strategy by which supporters, organizations, and other library data can be managed in the database reliant on bearings given by an external component, for instance, a provisioning subsystem.

Install server – as the name suggests, handles the custom establishment (installation) for an administrator's system.

The Robot Framework IDE (RIDE) is the coordinated improvement condition to complete automated tests for the Robot Framework. The components involved in RIDE includes:

- Test cases
- Test suite
- Keywords
- Libraries

- Resources
- Documentation
- Set-up
- Teardown

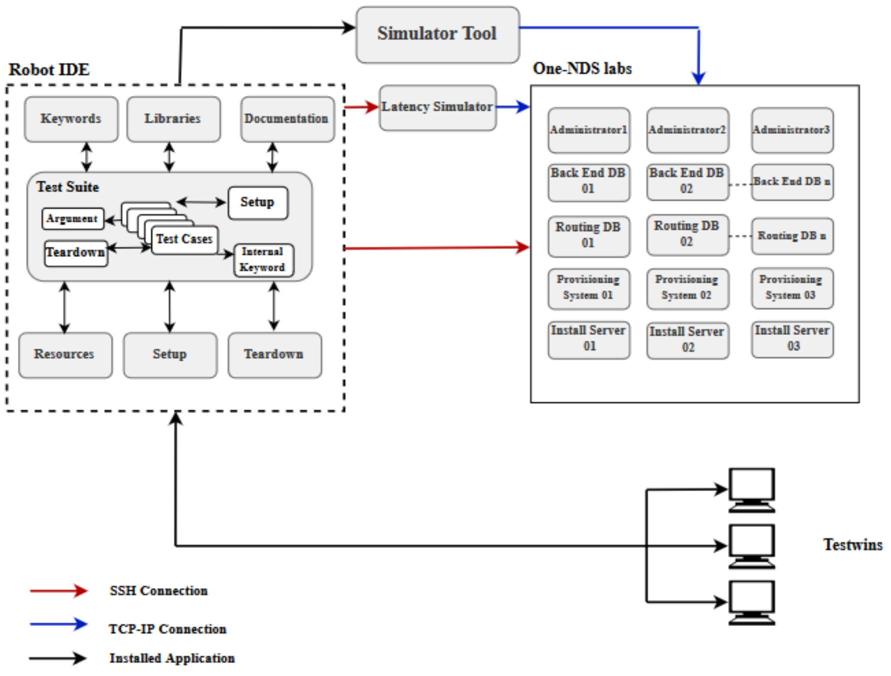

Fig 2. The design

CONCLUSION

Higher output and increased productivity have been two of the biggest reasons in justifying the use of automation. Advantages commonly attributed to automation include higher production rates, reduce production cost and increased productivity, more efficient use of materials and better product quality. The structure has helped us accomplish quicker execution of test cases without manual mediation. Better usage of assets by running tests medium-term, and manual tests being performed amid day time. Mechanization is exact and repeatable. Innovation that assistance you oversee work processes, automate excess assignments, give predictable experience to every one of your clients will enable you to give better dimensions of administration than your clients – and help improve your main concern.

REFERENCES

1. M. Fewster and D. Graham, Software Test Automation: Effective use of test execution tools. Addison–Wesley, 1999.

2. Liu Jian-Ping, Liu Juan-Juan, Wang Dong-Long, School of Information Science and Technology, “Application Analysis of Automated Testing Framework Based on Robot” Zhejiang Sci-Tech University, Hangzhou, China
3. David Barrett,“Automating Testing - Saving Time and Money” VLE Application Manager, E-Learning Development Team, University of York. 2013
4. Harsha T and B A Sujatha Kumari,“Software Test Automation with Robot Framework” in International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169, 5(6).
5. Bhuvaneshwari and Hemanth Kumar A.R, “Automation for Configuration of Mobile Networks using Robot Framework” , in, IJSDR , June 2017; 2(6)
6. P. Hsia, J. Gao, J. Samuel, D. Kung, Y. Toyoshima, and C. Chen, “Behavior-based acceptance testing of software systems: a formal scenario approach,” in Computer Software and Applications Conference,1994. COMPSAC 94. Proceedings., Eighteenth Annual International. IEEE, 1994; 293–298.
7. J. Itkonen, M. V. Mantyla, C. Lassenius, "How Do Testers Do It? An Exploratory Study on Manual Testing Practices", Proc. Third Int’l Symp. Empirical Software Eng. and Measurement (ESEM ’09),2009; 494-497
8. Myers, G.J., The Art of Software Testing, New York: John Wiley & Sons, 1979.
9. Beizer, B., Software Testing Techniques, New York: Van Nostrand Reinhold, 1990.
10. Mandara Nagendra, C. N. Chinnaswamy, T. H. Sreenivas, "Robot Framework: A boon for Automation", IJSDR, November, ISSN:2455-2631, 2018; 3(11): 446 – 449.
11. T. L. Graves, M. J. Harrold, J. Kim, A. Porters and G. Rothermel, "An empirical study of regression test selection techniques," Proceedings of the 20th International Conference on Software Engineering, Kyoto, Japan, 1998; 188-197.