ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 5, Issue 11, November 2017

Comparative Study and Analysis of Manual v/s Automated Product Feature Testing

Noorin Farnaz[1], Anjan K Koundinya[2]

Department of Computer Science Engineering, RV College of Engineering, Mysore Road, Bengaluru, India[1]

Department of Computer Science and Engineering, RV College of Engineering, Mysore Road, Bengaluru, India[2]

ABSTRACT: This paper does a comparative analysis of Manual and Automated testing on a Network Security Device. It highlights the advantages of automated testing through various comparison graphs and figures. In Manual testing, the test engineer creates test cases and executes manually to identify defects in the software. If the manual testing is repetitive, it is difficult to perform on large software applications which run on the hardware. This paper puts light on industry trends and lessons learned to save time and resources. Automation testing is done on the Product feature by selecting important test cases called sanity tests that cover the affected changefor a new build.

KEYWORDS: DUT(Device Under Test), ROI (Return on Investment), QA (quality Analysis), YAML, Sanity, JIRA, PCAP (Packet Capture), Time to Market

I. INTRODUCTION

Any productwhich is built or developed by an engineer or an enhancement on an existing product, needs to be tested. The testing ofany such product featuresshould be very intrusive, stringent and detailed, with no chance of errors. All these critical features constituting product need to be efficient and trustworthy to help gain importance in market. To ensure this, all the functionalities should be tested rigorously. There are two modes of testing based on the way testing is executed [1].

a. Manual
b. Automated

Manual testing is time-consuming and tedious, requires a lot of investment in human resources.With only manual testing, there is a risk of taping out major defects due to manual errors. A Device under Test (DUT) must be regressed thoroughly and sanitized. Automation not only helps in reducing the time needed for test cycle, but helps QA to qualify the product in important areas, as automation scripts does the major testing coverage. The time saved in running automated tests can be applied for doing ad-hoc and other tests.

Automation framework supports configuration of DUT, verification and Data collection scripts. Test scripts are specific to product features and consist of various test cases automated for that feature which can be run on any version of Device under test. Automated test scripts are very important in terms of economic factors, since it aims to completely replace manual testing which incurs payed manpower and tools. It helps a project complete on time which otherwise could lead to a significant business loss [2].

II. RELATED WORK

In [3], overcoming these shortcomings of manual testing by the benefits of automation testing is briefly mentioned. Automated testing is cost-effective, repeatable, reusable, programmable and comprehensive, more reliable and provides a better code-coverage in contrast with manual testing.Comparison of these forms of testingis done by taking a functional Features of an advanced network security product and performing its testing using manual and automated test methods.

DOI: 10.15680/IJIRCCE.2017. 0511007

A. Manual Testing ofNetwork Security Product:

Manual Testing is done on the network security system that is usually deployed in the enterprise network to protect the target network from any kind of Malware attack. Regression testing is done by the QA team on the latest available product build before deployment [4]. This kind of testing is very important for any software development or enhancement of the software application. Figure 1 shows the steps adopted for testing a single product feature on top of Linux Box or DUT(Device Under Test). Manual testing is usually done with the help of tool support such as Pcap Library, IXIA, Breakpoint, Git, JIRA, etc. As shown in the figure, all the steps need to be carried out by hand including data generation and test result validation.

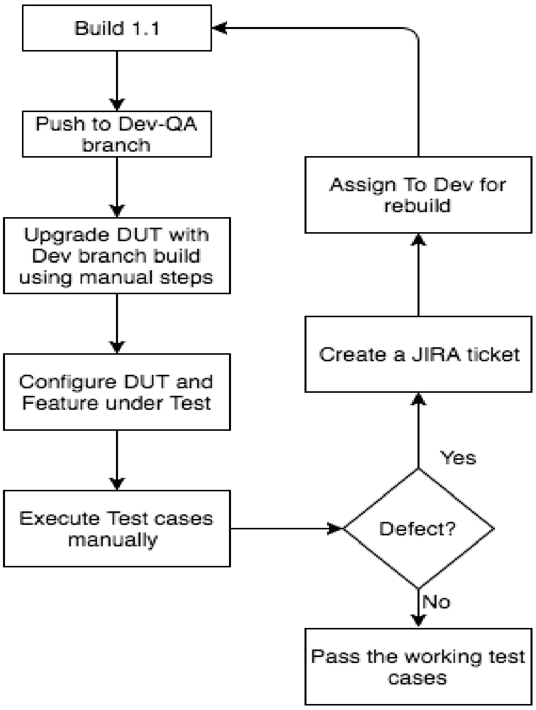
Build 1.1
Push to Dev-QA branch
Upgrade DUT with Dev branch build using manual steps
Configure DUT and Feature under Test
Execute Test cases manually
Defect?
Yes
No
Create a JIRA ticket
Assign To Dev for rebuild
Pass the working test cases

Figure 1: Manual-Testing Flow diagram

Whenever a build, for example Build 1.1 which is the next version of Build 1.0 is available for testing, any extra change affecting the previous features need to be thoroughly tested for any defects. This build is made available using Git repository, wherein a separate Dev-QA branch is created and the code is pushed in [5]. Above process is shown for one feature under test. Depending on the number of affected features and the number of product modelsto be tested, it requires a great deal of effort for the testing team to do it manually.

B. Automation Testingof Network Security Product:

Automated testing is nothing but executing a series of actions for any test validation without any Human intervention. It can also be defined as testing the new software build again and again seamlessly without the tester having to worry [6].Proposed Automated Testing dramatically reduces all the manual testing overhead and speeds up the testing process. To build the automation test scripts for CLI based feature validation, Python unit testing Framework is used, and for the Web UI test validation, Selenium Web driver is used to derive different metrics considered in the later sections. Figure 2 shows the flow of testing done through automation with least human intervention. PyUnit framework is made use of, to code the test suites for individual feature. Each typical sanity test case mainly consists of setup, enabling feature under test on the linux box, executing pcap for target Malware stored in the Replay server, validating the alerts generated against the values in Database. YAML file is used to specify configuration specific information.

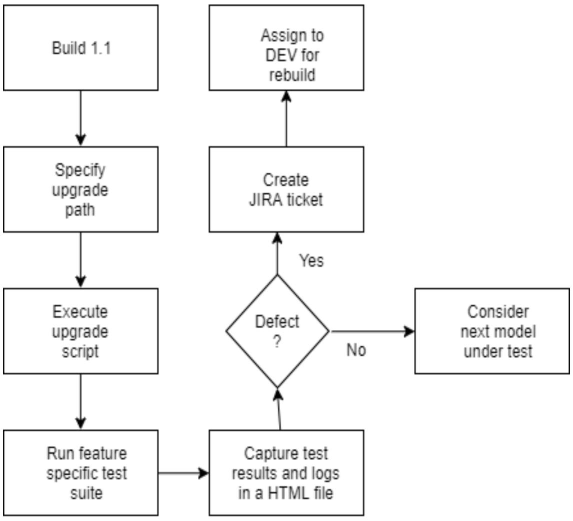
Build 1.1
Specify upgrade path
Execute upgrade script
Run feature specific test suite
Capture test results and logs in a HTML file
Defect ?
Yes
No
Consider next model under test
Create JIRA ticket
Assign to DEV for rebuild

Figure 2: Automated-Testing Flow diagram

The IDE used for development of test suites isPycharm along with versioning support from Git Hub. Python is the programming language for Automation development. Above diagram shows the steps taken by a test run on the Linux controller for a functionality. Time taking tasks like upgrading and rebooting the box to the latest build, running the feature specific test cases and producing a log report is all done automatically in a very less time [7]. Once the HTML report is available to the tester, the failed test cases can be analyzed and the JIRA ticket can be created against the missing gaps.

III. METRICS AND RESULTS

A. Evaluation Metrics:

Evaluation Metric is used to specify measurement of anattribute of a software project. Test activities concerning the test activities is produced by QA Metrics. Testing Metrics help in gaining quantitative insight into the effectiveness of software testing process [8].

Table 1 describes various metrics along with their values that are used for getting graphs in the following sections.These metrics are drawn for only five features of the product to make the comparison graphs simpler. There are hundreds of such features tested on a product before the build is released to the real time Devices.

Table 1: Manual Testing Metrics with sample values.

Feature Type	Feature Name	No. of Test Cases	Manual Testing Effort (in hours per feature)	Test Results with Human Errors
Basic	Feature_001	22	12	5
Sanity	Feature_002	116	56	23
Sanity	Feature_003	076	44	20
Critical	Feature_006	148	80	45
Critical	Feature_005	109	68	33

DOI: 10.15680/IJIRCCE.2017. 0511007

Above table describes different types of features along with the respective number of test case and testing efforts. The same metrics are considered to derive values for automated testing process. Table 2 captures these details for Automated form of testing. From the table itself, it is evident in how significant the difference is in these two testing processes with respect to time.

Table 2: Automated Testing Metrics with values.

Feature Type	Feature Name	No. of Test Cases	Manual Testing Effort (in hours per feature)	Test Results withAutomation Errors
Basic	Feature_001	22	0.6	2
Sanity	Feature_002	116	1.5	5
Sanity	Feature_003	076	1.2	3
Critical	Feature_006	148	3.0	7
Critical	Feature_005	109	2.4	12

Apart from the metrics mentioned in the tables, there are other non-feature specific metrics that are important to consider [9]. They are:

Product Models Under Test: Each feature in question for a buildneeds to be tested for different product models. In the case study, it is for network security product, which has various hardware models and versions functioning in the Market.

Time to Market:Amount of time it takes for a Project from the time it starts till the time it is deployed to the customer environment.

Defect validity:Reliable defects that are raised post testing of different product features.

Return on Investment: ROI can be defined as amount of net gain from a project release after subtracting the cost of investment. Automated way of testing requires lesser manpower hence increases the ROI by decreasing the cost.

B. Comparison Results of Manual and Automated Testing Process

Testing metrics provide a very valuable insight to the testing process. In this section, we consider two differentiating metrics, first one is in terms number of test cases executed in every 10-hour time slot manually and through automation. The second comparison is done in terms of reliability measure of manual and automation testing process with the help of defect acceptance rate.

1. Manual versus Automated test cases Covered in every 10-hour time duration:

First comparison can be done based on the time taken by the number of manual testsrun relative to number of tests to be coded and run using automation in a specific time slot. This statistic is considered for the first test run, for a single product model and two testers. Values are obtained at a fixed time slot difference of 10 hours. Figure 1 shows a graph with three variables for the values mentioned in the table. X-axis denotes the time slot duration in hours and Y-axis denotes the number of test cases run.

Table 3, shows the values for number of test cases that are run to produce results manually versus test cases covered by automation scripts in the same duration.

Table 3: Manual vs Automation test run values

Duration of Test run (in hours)	Manual Test Cases	Automated Test cases
10	25	60
20	40	110
30	60	260
40	85	350

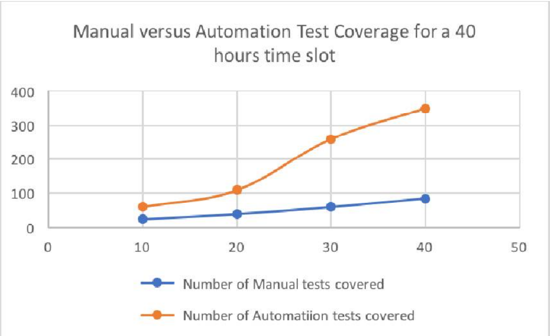

Figure 1: Manual v/s Automation Test Run Graph

From the above graph,we can see a significant difference in the manual and Automation test run data. It clearly shows the shows the number of automated and manual test cases that testing can possibly accomplish, given the available budget and a time slot.With this graph, we can derive the efficiency of automated test runs with a trade-off in testing phase.

2. Defect Acceptance Rate of Manual and Automated Testing:

The test engineer determines number of valid defect during test execution. To get better picture we can compare the previous build(s) values. Defect Acceptance = (No. of Valid defects /Total No. of defect)*100.Table 4, shows the values for number of defects in each product builds, valid manual testing defects, valid automation testing defects, Acceptance Percentage of Manual defects and Acceptance percentage of Automation defects scripts with respect to each build. Here we assume, the total number of defects are same for manual and automation runs for the purpose of comparison [10].

Build	Total Defects	Valid Manual Testing Defects	Acceptance %	Valid Automation Testing Defects	Acceptance %
1st	80	45	56	66	82
2nd	84	46	55	75	89
3rd	60	40	67	50	73

Table 4: Defect Acceptance Percentage of Manual and Automation testing across Builds

Graphs plotted considering the values in the above table are shown below with the comparison study of different defect acceptance percentage in various product builds. Defect acceptance can be defined as number of defects raised by the QA team which was accepted by the development team for evaluation ad changes [11].

Figure 2 represent comparison of Defect Acceptance values of different builds. Table (3) shows the defect acceptance for various release. Bar graph on the left-hand side denoted the values plotted for Defect Acceptance percentage for manual testing. It is 56% than compared to the acceptance percentage of the automation testing defects which leads with 82% acceptance. Similarly, 2nd and the 3rd builds or releases have 55% and 67% respectively, when compared to 89% and 73% in automation acceptance.

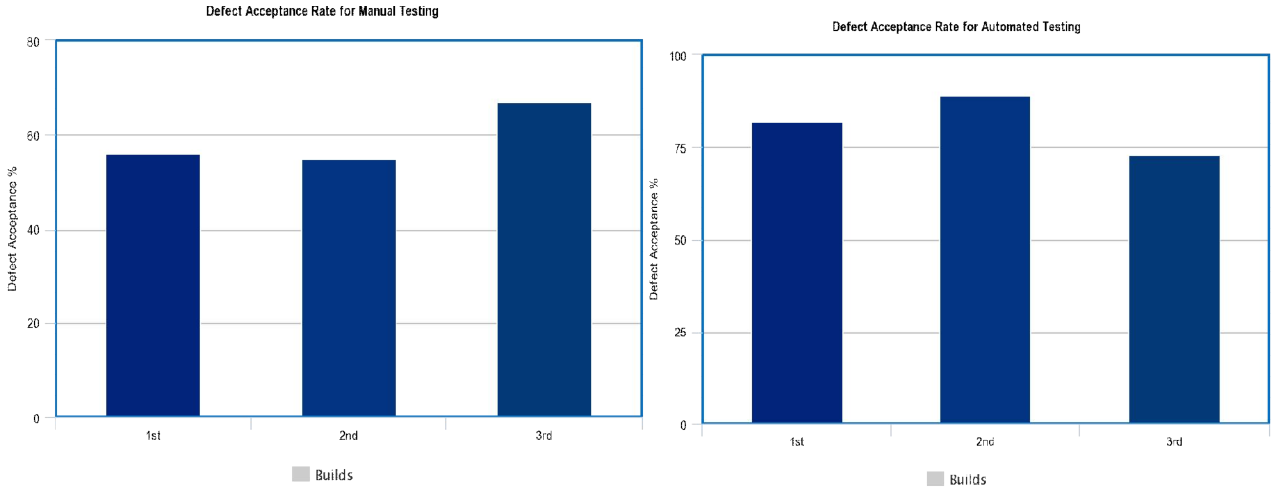

Figure 2: Defect Acceptance Percentage Graphs for Manual and Automation testing

From the above graph, we can see that the defect rejection rate is higher in manual testing because of various manual errors than compared to defect rejection rate in automated testing. From this analysis, reliability of automation testing can be studied. In a software application system, although not always 100% of the testing can be automated, automation makes a significant difference in time and effort of testing even if at least 80% of them are automated [12].

IV. CONCLUSION AND FUTURE WORK

In this paper, comparison of manual and automated ways of testing is done using various metrics. For a software project to be built and deployed for any product, there is a challenge of producing tested quality products by combining and balancing automated and manual testing. Different tables and graphs are used to show how significantly automated testing makes a trade-off with the manual testing metrics. Here, only limited number of metrics is used which can be further explored in the future works. Various other metrics like time to market, efficiency of tester is not considered, metrics for combined manual and automation testing can be further explored.

ACKNOWLEDGEMENT

With an immense pleasure and with a deep sense of gratitude, I convey my heartfelt and sincere thanks to my guide **Dr.Anjan KKoundinya, Professor, Department of Computer science and Engineering, R.V.C.E., Bengaluru.**

Dr.Anjan KKoundinyawould like to thank lateDr.V. K AnanthaShayana,Former Head, Dept. Of CSE,MSRIT, Bengaluru for igniting the passion for research.

REFERENCES

1. R. M. Sharma, "Quantitative Analysis of Automation and Manual Testing", *International Journal of Engineering and Innovative Technology (IJEIT)*, Volume 4, Issue 1, Jul-14, pp. 23-29
2. Juki Wirawan Tantra, "Experiences in Building Python Automation Framework for Verification and Data Collections", *The Python Papers Monograph 2:17 Proceedings of PyCon Asia-Pacific*, 2010, pp. 173-176.
3. Eduard Enoiu, Daniel Sundmark, Adnan _auev, "A Comparative Study of Manual and Automated Testing for Industrial Control Software", *Software Testing, Verification and Validation (ICST)*, 2017 IEEE International Conference, ICST.2017.44, Mar 2017, pp. 1-2

4. R. M. Sharma, "Quantitative Analysis of Automation and Manual Testing", *International Journal of Engineering and Innovative Technology (IJEIT)*, Volume 4, Issue 1, Jul-14, pp. 23-29
5. XiaoliLian, Jane Cleland-Huang, Li Zhang, "Mining Associations Between Quality Concerns and Functional Requirements", *Requirements Engineering Conference (RE)*, 2017 IEEE 25th International, ISSN:2332-6441, 2017, pp. 34-37
6. Muhammad Masroor Ali and Tanay Kumar Saha, "A Proposed Framework for Full Automation of Software Testing Process", *International Conference on Informatics, Electronics & Vision*, 2012, pp. 1-2
7. Dmitry Tishchenko, "Defining Test Case Automation Metric", *Software testing Magazine*, Apr 2014, pp. 1-2
8. DainalHooshyar, Rodina Binti Ahmad, MohdHairulNizamMdNasi, Wong Ching Mun, "Flowchart-based Approach to Aid Novice Programmers: A Novel Framework", International Conference on Computer and Information Sciences (ICCOINS), 2014, pp. 3-4
9. Hitesh Tahbildar and BichitraKalita, "AUTOMATED SOFTWARE TEST DATA GENERATION: DIRECTION OF RESEARCH", *International Journal of Computer Science & Engineering Survey (IJCSES)*, Vol.2, No.1, Feb 2011, pp. 99-120
10. Muthukumar S, Dr.Krishnan N, Pasupathi.P, Deepa. S, “Analysis of Image Inpainting Techniques with Exemplar, Poisson, Successive Elimination and 8 Pixel Neighborhood Methods”, International Journal of Computer Applications (0975 – 8887), Volume 9, No.11, 2010
11. Shruti N. Pardeshi, "Study of Testing Strategies and availableTools", *International Journal of Scientific and Research Publications*, Volume 3, Issue 3, Mar 2013
12. Johnson Selwyn, Ambica Jain, Whitepaper on Test Automation Framework using MBT, *HCL Technologies*, 2013, pp. 4-7