Survey on Software Testing

Abstract

Software testing is an integral part of software engineering referred to high quality software development as a result it is essential to use software testing methods and tools effectively and efficiently. In this survey, I'm analyzing functional and nonfunctional requirements testing and studied 30 papers related to the domain. Much of the research is done on functional testing but least in nonfunctional requirement testing. This paper also provides the comparison of both testing efficiently. On the other hand, it also provides summary of 30 papers that are published in ACM, IEEExplore and Science direct. There are a number of reported techniques for testing the software to specify the requirements before releasing. This paper includes the identification of current practices and opportunities for improvement in testing, and classify various approaches for testing.

1. Introduction

In the field of software development code must adhere to various functional and non-functional requirements, such as architectural requirements. These requirements are then gathered and confirmed by signing of noncompliance agreements, via written documents. The cost in having to communicate the non-functional requirements is high due. This is because all of non-functional points need to be conveyed and need to understand for all stakeholders. The architectural requirements play a vital role in the life of a software and the maintainability later on as the architecture sets the tone for code reusability, and cost efficiency in maintaining later versions and updates of the software.

The standards of the requirements are then confirmed by creation of multiple reports. Best case scenario is where these guidelines are located at a central point but often times, they are distributed in different document systems. Rules regarding organization, software's and or operating systems are spread everywhere. Rarely these guidelines are found accumulated in one spot. The research process, is planned by keeping view of the objectives. The process consists of 5 main steps, first is systematic mapping. This step was executed to describe the NFRs testing approaches on mobile applications. In addition, this step attempts to analyze the level of automation. Software testing on mobile apps is all different as per the nature of the mobile app and the specific platform it runs on. The app could be native, hybrid, developed on web.

Resource management like energy and memory or execution time are more or less correlated strongly with different sections of the codebase. This causes the testing to be targeted at sections of code instead of the entirety of the source. The targeted mutation analysis is applied directly to the parts of the code where execution time is impacted significantly is not attended in an efficient manner. In mutation analysis, multiple copies or versions are made systematically by applying rules for changing the syntax, elements, and mutation operators.

Understanding Non-functional Intentions via Testing and Experimentation (UNITE), is created to unit test non-functional requirements of three component-based distributed systems. The results acquired after applying the tool UNITE displays how it simplifies unit testing and evaluation of non-functional properties during the early stages of the lifecycle of the software.

Data analysis and aggregation is the process of evaluation extracted data based on a user-defined equation and combining multiple

results to a single result. QED's goal is to provide reliable and real-time communication that is able to withstand the dynamic changing environment of GIG environments.

The process of gathering is necessary to unit testing since it operates on a single piece of result to determine if it passes o fails. Focus is on making crucial information readily available. It is noted that there is a tendency for teams to forget or avoid repetitive and cumbersome tasks. Some of these common tasks are reading application logs from multiple servers. Another issue is that test and production environments must be compatible as well as security policies. To allow this to happen costs and operational routines must allow this to occur.

2. Classification on the basis of websites and application

New method of Constraints and Object oriented exist, so that design decisions are captured in goal graph and reused by other models. Classification may be tending towards functional and non-functional constraints that are beyond the traditional taxonomy of requirements that allows classification more precisely and tested effective accordingly. All different types of development would run on their own platforms using their respective defined software. Since this is the case with mobile applications, they need to be tested differently with tools to ensure quality is maintained in functions, performance, behaviors, services, as well as attributes like usability, interoperability, connectivity, security, privacy and many more.

The main issue of web applications is load and performance testing. To ensure their performance ability and load testing, web application should be able to answer user's request in appropriate time and solve other user's issues simultaneously. Secondly, it should not have broken links and also it must be able to navigate to the whole website from the homepage.

The paper proposes a framework for the process of testing process where the non-functional requirements that should be considered for mobile application tests, test case scenarios (TCS) and tools to test and execute these TCS. They allow to control and ensure the quality of the product. This results as a cost-benefit analysis according to the types of test execution, the validation and execution of the proposed framework that is shown.

Mobile application testing imposes several challenges and peculiarities, such as limited power, interface adaptation, and user's privacy. The current state of MT technique is being conducted to identify the main problems faced to establish testing strategies for the context of NFRs. Through Systematic mapping the process of data analysis, and some information related to security can be observed. This aims to automatize the verification of compliance of NFRs on mobile applications through MT. Neglecting such requirements is costly because difficult to fix the system if it has already been implemented.

3. Comparison Of Approaches

In study, I have discussed various approaches that functional testing requires writing tests which verify all three types of constraints. This means that every class attribute, relationship and method should be checked to ensure that it conforms to appropriate constraints. At this stage, providing good project documentation is essential as it represents a formal specification of all constraints.

In practice, the validation of the functional and non-functional requirements is not given

the same importance. Several techniques are presented that enable cost effective validation of non-functional requirements. Approach is targeting validation of performance properties. Cost effectiveness of performance validation by automatically generating load tests and subsequently introduced a compositional load test generation technique that targeted more complex software systems. Robustness and cost effectiveness of such validation is improved by using exception-handling constructs and by amplifying existing tests to exhaustively test every possible pattern in which exceptions can be raised respectively.

Testing starts from requirement elicitation to design and code phase to lower down risk failure. Testing requirements in an industrial software system is a unique challenge. In order to try a different approach, the study will be using aspect orientated techniques to test for requirements. Test case scenarios and tools applied to test and execute these cases. They allow to control and ensure the quality of the product. This results as a cost-benefit analysis according to the types of test execution, the validation and execution.

Quality models classifies qualities that leads towards requirements. Testing determines the choose attributes fulfilled the system requirement effectively. Load and performance testing ensure their performance ability in applications while web applications should be able to answer user's request in appropriate time and solve other user's issues simultaneously. Search based technique identifies the size and complexity of the software. This technique shows the metaheuristic search technique for non-functional execution of time, security, safety, unitability and quality of service.

Non-functional requirements are a challenge to incorporate and test. In agile development, it is required to cater these requirements. Due to the nature of these, identifying factors that influence the testing in performance and security aspects will be empirically sought out by interviewing IT professionals. This will enable to get a sense of which factors to look out for and how to handle them.

Open-source toolkit enables automated testing of requirements that offers an alternative to current approaches testing suites by being flexible and provides developers as well as project managers with under development systems reports. Automated execution of system's program in Server in order to exercise the system as a whole in a realistic manner. Gathering a system behavior description is helpful and also simplifies matching error messages in logs with other reported system abnormalities. The archive of reports makes it possible to track the system behavior over time.

4. Conclusion

Software testing tools are being used in a limited way. Tools usage was limited to activities such as recording bugs and test results. Tools are rarely used in defect prevention due to its high complexity and difficulty. Though defect prediction is a popular research area on which much research has been conducted, there are practical problems that fault data are not properly manipulated in most projects and that the cost to collect and maintain fault data is too high. The defect pattern discovery must be improved so as to cope with such problems.

Model Based Testing (MBT) is a new-age test automation technique with the capability of generating test cases by using model developed from the analysis of the abstract behavior of the System.

In software development much of the work is done in a tool-chain that contains tools like

text editors, compilers, linkers, static analysis tools, automatic test frameworks. The architectural requirements often control the ability of the software to be reused and maintained in a cost-efficient way. When a developer is writing code, they often focus on implementing one functional requirement at the time. This requirement can be verified using automated tests and is often located in an isolated part of the software. The tool-chain analyzes and transforms of the code in many different ways and the tools can give fast feedback to the developer about the progress. This means that the tool-chain can be used to prevent mistakes from the developer.

References

[1] *C K. Dyrkon and F. Wathne, "Automated testing of functional requirements," Proc. Conf. Object-Oriented Program. Syst. Lang. Appl. OOPSLA, pp. 719–720, 2008, doi: 10.1145/1449814.1449828.*

[2] *J. H. Hill, H. A. Turner, J. R. Edmondson, and D. C. Schmidt, "Unit testing nonfunctional concerns of component-based distributed systems," Proc. - 2nd Int. Conf. Softw. Testing, Verif. Validation, ICST 2009, pp. 406–415, 2009, doi: 10.1109/ICST.2009.44.*

[3] *M. Felderer, B. Marculescu, F. Gomes De Oliveira Neto, R. Feldt, and R. Torkar, "A testability analysis framework for non-functional properties," Proc. - 2018 IEEE 11th Int. Conf. Softw. Testing, Verif. Valid. Work. ICSTW 2018, pp. 54–58, 2018, doi: 10.1109/ICSTW.2018.00028.*

[4] *B. Lisper, B. Lindstrom, P. Potena, M. Saadatmand, and M. Bohlin, "Targeted Mutation: Efficient Mutation Analysis for Testing Functional Properties," Proc. - 10th IEEE Int. Conf. Softw. Testing, Verif. Valid. Work. ICSTW 2017, pp. 65–68, 2017, doi: 10.1109/ICSTW.2017.18.*

[5] *F. L. Arteaga, F. R. Contreras, and A. Barrientos, "A framework for nonfunctional testing process of mobile applications," Proc. 2019 IEEE 26th Int. Conf. Electron. Electr. Eng. Comput. INTERCON 2019, pp. 1–4, 2019, doi: 10.1109/INTERCON.2019.8853813.*

[6] *C. R. Camacho, S. Marczak, and D. S. Cruzes, "Agile team members perceptions on non-functional testing influencing factors from an empirical Study," Proc. - 2016 11th Int. Conf. Availability, Reliab. Secur. ARES 2016, pp. 582–589, 2016, doi: 10.1109/ARES.2016.98.*

[7] *R. Lagerstedt, "Using automated tests for communicating and verifying nonfunctional requirements," 2014 IEEE 1st Int. Work. Requir. Eng. Testing, RET 2014 - Proc., pp. 26–28, 2014, doi: 10.1109/RET.2014.6908675.*

[8] *F. L. Arteaga, "A Framework for Non-Functional Testing Process of Mobile Applications," 2019 IEEE XXVI Int. Conf. Electron. Electr. . Eng. Comput., pp. 1–4, 2019.*

[9] *C. R. Camacho, S. Marczak, and D. S. Cruzes, "Agile Team Members Perceptions on Non-Functional Testing Influencing Factors from an Empirical Study," pp. 582–589, 2016, doi: 10.1109/ARES.2016.98.*

[10] *M.Glinz, "Rethinking the Notion of Non-Functional Requirements," no. September, pp. 55–64, 2005*

[11] *W. Afzal, R. Torkar, and R. Feldt, "A systematic review of search- based testing for non-functional system properties," Inf. Softw.Technol., vol. 51, no. 6, pp. 957–976, 2009, doi:10.1016/j.infsof.2008.12.005.*