Software testing and its types information technology essay

Technology, Information Technology

ABSTRACT

In this paper we discuss what software is testing, what are its types and how they are performed? Software testing is an important part of software development life cycle and it consumes a lot of time. But it ensures correctness, completeness and quality of developed computer software. This paper explains techniques of software testing and various sub types of both functional and non-functional testing. We cover unit testing, integration testing, smoke testing, system testing, regression testing, user acceptance testing, performance testing, scalability testing, compatibility testing, data conversion testing, security testing and usability testing. As testing is a time consuming activity we also discuss the ways of increasing efficiency and productivity in software testing. We discuss how the manual testing and automated testing activities are performed with the help of testing tools; one such tool is HP business process testing software. We discuss how this tool is going to be helpful in near future. KEYWORDS : Software testing, unit testing, integration testing, smoke testing, system testing, regression testing, user acceptance testing, performance testing, scalability testing, compatibility testing, data conversion testing, security testing and usability testing, Manual testing, automated testing, HP business process testing software, test scripts, components, subject matter expert, QA (Quality Analyst) and ROI (return on investment).

INTRODUCTION

Testing is a very important activity in product development lifecycle as it measures the quality of product and helps in determining production readiness of an application. It checks whether all requirements are implemented correctly and detects non-conformances if any, before deployment. Testing makes software predictable in nature, improves quality and reliability. It also helps marketability and retention of customers. The various factors that contribute to making testing a high priority of any software development effort include: Reduction of software development cost - Testing software in the initial stages of development reduces the cost of developing the program. A problem that goes undetected in the initial software development lifecycle stages can be much more expensive to resolve at a later stage. Ensures completeness of the product - Testing a software product ensures that the customer requirements map to the final product that is delivered. Reduction in total cost of ownership- By providing software that looks and behaves as shown in the user documentation, customers require fewer hours of training support from product experts and thus reduce the total cost of ownership. Accretion of revenues - A bug free code (which is obtained only after intensive testing) also brings in customer satisfaction, which leads to repeat business and more revenues.

WHAT IS TESTING?

" Testing is an activity in which a system or component is executed under specified conditions; the results are observed and recorded and an evaluation is made of some aspect of the system or component" – IEEE. Software testing is a process used to identify the correctness, completeness and quality of developed computer software. It includes a set of activities conducted with the intent of finding errors in software so that it could be corrected before the product is released to the end users.

WHY TESTING?

It is the primary duty of a software vendor to ensure that software delivered does not have any defects and that the customer's day-to-day operations do not get affected. This can be achieved by rigorously testing the software. To protect an organization from any trouble and in order to address various risks involved during a change to an organization, testing is important. Risks can be related to the ones affecting reputation or resources or could be the ones leading to legal issues. All the above incidents only reiterate the significance of thorough testing of software applications and products before they are pushed to production. It clearly demonstrates that cost of rectifying defect during development is much less than rectifying a defect in production.

TESTING TECHNIQUES

Software can be tested either by running the programs and then verifying each step of its execution against expected result or by statically examining the code against the stated requirements. These 2 distinct methods have led to the popularization of 2 techniques viz. Static Testing & Dynamic Testing:

1. 1STATIC TESTING

This is a non-execution-based testing technique. It could be done during any phase in the software development lifecycle but largely it occurs during requirement, design and coding phase. The Design, code, test plan, test cases or any document may be inspected and reviewed against a stated requirement/ standards/some guidelines/checklists. Static Testing includes: Informal Reviews – These reviews are generally done by a peer and occur on a need basis. Walk-through – Semiformal review facilitated by the author of the product. Inspections – Formal reviews facilitated by a knowledgeable person who is not the author of the document. Many studies show that the single most cost-effective defect reduction process is the static testing - the code inspection/walk-through/reviews.

Advantages of static testing :

Capture defects early, so saves costChecklist-based approachFocuses on coverageHighest probability of finding defectsEfficient way to educate people regarding the productIndependent of test environment setups

Disadvantages of static testing :

Time consumingCannot test data dependenciesHigh skill levels required

1. 2DYNAMIC TESTING

This is an execution-based testing technique. Here, the program, module or the entire system is executed and the output is verified against the expected result. Dynamic execution of tests is based on one of the following: Dynamic testing can be further classified into White Box Testing and Black Box Testing

1. 2. 1WHITE BOX TESTING

this testing technique takes into account the internal structure of a system or component. Complete access to the object's source code is needed for white-box testing. This is known as ‘ white box’ testing because tester gets to see the internal working of the code. White box testing helps to: Achieve high code coverageTest program logicEliminate redundant codeTraverse complicated loop structuresCover control structures and sub-routinesEvaluate different execution pathsUnit testing and some part of integration testing fall under white box testing category.

1. 2. 2BLACK BOX TESTING

a testing method where the application under test is viewed as a black box and the internal behavior of the program is completely ignored. Testing occurs based upon the requirement specifications. Black box testing is conducted more from a user’s perspective. It focuses on the features and not the implementation. Provides a big picture approach. Black Box testing technique can be applied once unit and integration testing are completed. System testing and regression testing fall under black box testing category.

Advantages of dynamic testing:

White box testing:

Logic of the system is testedThose parts which could have been omitted in black box testing are also getting covered. Redundant code eliminatedCost effective when appropriate techniques are used

Black box testing:

Simulates actual system usageMakes no assumptions about the system structure

Disadvantages of dynamic testing:

White box testing:

Does not ensure that all user requirements are metMay not simulate real-time situationsSkill level needed is high

Black box testing :

May miss out logical errorsChances of redundant testing is thereCannot decide which part of code is not getting executed. Thus a good combination of black box and white box testing can ensure adequate code, logic, functionality coverage.

TYPES OF TESTING

Testing could be classified at a high level as Functional testing and Non-functional testing. Chart below is a snapshot of the different types of testing.

2. 1FUNCTIONAL TESTING

Functional Testing refers to verifying if the module performs its intended functions in accordance with the specification. The purpose is to ensure that the application behaves in a way it is expected to behave E. g. data entry, navigation, processing, retrieval and display based on requirements. Different types of functional testing are explained below.

2. 1. 1UNIT TESTING

Primary test performed on software is the ‘ Unit Testing’ to see if a standalone module is working as per the requirements. Testing performed on a single, standalone module or unit of code to ensure correctness of the particular module. Focuses on implementation logic, so the idea is to write test cases for every method in the module. The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. This isolated testing provides four main benefits: Flexibility when changes are required. Facilitates integrationEnsures documentation of the codeSeparation of interface from implementationThis type of testing is mostly done by the developers.

2. 1. 2INTEGRATION TESTING

Phase of software testing in which individual software modules are combined and tested as a group. It follows unit testing and precedes system testing. It takes the unit tested modules as input, groups them in larger aggregates, applies tests defined in an Integration test plan and delivers as its output, the integrated system which is ready for system testing. Data transfer between the integrated modules is thoroughly tested. Dummy modules interface viz. Stubs and Drivers are used in integration testing. Drivers are simple programs designed specifically for testing the calls to lower layers. It provides emerging low-level modules with simulated inputs and the necessary resources to function. Stubs are dummy software components used to simulate the behaviour of a real component. They do not perform any real computation or data manipulation. It can be defined as a small program routine that substitutes for a longer program, possibly to be loaded later or that is located remotely.

Two methods of integration are :

IncrementalBig bang

Incremental:

It involves adding unit tested modules one by one and checking each resultant combination. This process repeats till all modules are integrated and tested. Correction is easy as the source and cause of error could be easily detected.

Big bang:

Modules unit tested at isolation are integrated at one go and the integration is tested. Correction is difficult because isolation of causes is complicated.

Three strategies of integration are:

Bottom-Up Strategy:

Process starts with low level modules of the program hierarchy in the application architectureTest drivers are usedThe following diagram shows the integration of modules in case of Bottom-Up strategy.

Top-Down Strategy:

Starts at the top of the program hierarchy in the application architecture and travels down its BranchesStubs are used until the actual program is readyThe following diagram shows the integration of modules in case of top-Down strategy.

Sandwich Strategy

Integration of Top-Down and Bottom up methodInstead of completely going for top down or bottom up, a layer is identified in between. The following diagram shows the integration of modules in case of sandwich strategy

2. 1. 3SMOKE TESTING

This is a quick & non exhaustive test performed on the new version of the software to see whether it is performing well enough to accept it for major testing. This test is used to validate if the major functions of a piece of software work as intended.

Reasons, why a build could be rejected includeMajor functionalities are not working fine or missingNavigations are not appropriateLook and feel is not according to specification

2. 1. 4SYSTEM TESTING

Black-box type testing that is based on overall system requirement specifications. It is carried on an integrated system and end to end testing is carried out. . During system test execution phase, defects that can only be exposed by testing the entire system are found.

2. 1. 5REGRESSION TESTING

Testing conducted for the purpose of evaluating whether or not a change (defect fix or enhancement) to the system has introduced a new failure. This refers to continuous testing of an application for each new build.

2. 1. 6USER ACCEPTANCE TESTING

Acceptance testing is one of the last phases of testing which is typically done at the customer place. Generally users of the system will perform these tests which ideally are derived from the User Requirements Specification, to which the system should conform. The focus is on a final verification of the required business function in a simulated environment which is very close to the real environment. Idea is that if the software works as intended during the simulation of normal use, it will work just the same in production. These tests are often used to enable the customer to determine whether or not to accept a system. Planning for this should be done during the requirement analysis phase, which will help to identify the gaps in requirements and to verify the testability of the requirements. Acceptance testing will be carried out when the test team has completed the System testing.

Types of UAT:

Alpha Testing: Simulated or actual operational testing performed by end users within a company but outside development group. Beta Testing: Simulated or actual operational testing performed by a sub-set of actual customers outside the company.

Globalization Testing:

To ensure that internationally localized versions do not have problems unique to language/currency etc. To validate whether application developed provide support forMulti-languageMulti Currency

Localization Testing:

Subset of globalization testing and checks for a particular localeThis test is based on the results of globalization testing, which verifies the functional support for that particular culture/locale. Can be executed only on the localized version of a product

2. 2NON-FUNCTIONAL TESTING

Non-functional testing verifies if the application performs its intended functions as per the non-functional requirements which could be performance, security, usability, compatibility etc.

2. 2. 1PERFORMANCE TESTING

This testing is carried out to analyze/measure the behavior of the system in terms of time, stability and scalability and the parameters generally used are response time, transaction rates etc. This is done to verify whether the performance requirements have been achieved. Performance testing is implemented and executed to profile and " tune" an application's performance behaviors as a function of conditions such as workload or hardware configurations.

Types of Performance Testing :

Load Testing :

Load testing is defined as type of performance testing where performance of the application is monitored when subjected to different loads. Load is referred to as the number of users using the application. Load testing consists of simulating a real-time workload conditions for the application under test. Generally refers to the practice of modelling the expected usage of a software program by simulating multiple users accessing the program's services concurrently. To determine at what load the system fails or system’s response time degrades

Stress testing:

Stress testing is conducted to evaluate a system or a component at or beyond the limits of its specified requirements. Ideally, stress testing emulates the maximum capacity the application can support before causing a system outage or disruption. It ensures that the application which is tested for expected load, can withstand spikes in load conditions (like increase in rate of transactions). Based on the results of stress testing, system can be configured and fine tuned for optimal performance. This test determines the failure point of the system under extreme pressure. Useful when systems are being scaled up to larger environments or being implemented for the first time. System is monitored for performance loss and crashing during the load times.

Endurance Testing

Execute the test with expected user load sustained over longer period of time with normal ramp up and ramp down.

Volume Testing

Volume testing is the testing where the system is subjected to large volume of data to determine its point of failure. The main objective of volume testing is to find the limitations of the software by processing large amount of data.

2. 2. 2COMPATIBILITY TESTING

Test to validate that the application functions the same way across different supportedHardware and software configurationsOperating systems (OS)Web browsersDatabase types

2. 2. 3DATA CONVERSION TESTING

The Data Conversion testing is done to validate the Conversion of the source data to the target data. Data Conversion testing and implementation are practically inseparable. The Data Conversion testing plan should be made to confirm the following: Whether or not the source data type has been converted to the target data type? Is there any loss in the data? Is data integrity maintained?

2. 2. 4SECURITY/PENETRATION TESTING

Security testing evaluates that an Information Systems maintains confidentiality, availability and integrity of data. Security testing is performed to assess the sensitivity of the system against unauthorized internal or external access. Testing is done to ensure that unauthorized persons are not given access.

Special skills required for security testing

Ability to think like a hackerBeing aware of all known vulnerability & exploitsThorough understanding of runtime environmentIdentification of criticality and sensitivity of data assets

2. 2. 5USABILITY TESTING

In usability testing, software is evaluated for the easiness with which user can learn and use the application. Essentially it means testing the software to ensure that it is ‘ user friendly’ in nature.

3. FUTURE SCOPE AND CONCLUSION

Today many tools are available for both manual and automated software testing. Though automated testing is very helpful but still enormous amount of testing is done manually. Many companies have introduced tools for manual testing also, these tools decrease the amount of time and effort a tester has to spend in manual software testing. One such tool is HP business process testing software. Manual software testing process can be optimized using HP business process testing software. Manual software testing has been modernized using HP business process testing software. HP designed this product to get beyond the limitations of testing solutions. The sole idea is to help professionals who have no coding experience to quickly create test scripts/cases for the overall business process. This product simplifies and speeds up the test design process and accelerates time-to-deployment process when overall testing is completed at the end of cycle. This also increases the productivity of development and testing teams. This product also smoothens the transition process from manual testing to automated testing which improves overall business process of any organization. The concept of re-usability of business " components" for designing test cases was also introduced by this HP product which is called componentization. Also features like parameterization, iteration and maintenance have been focused on by HP to improve the process of manual testing. This product eased automation by providing different tools and software's to automate the componentized test definitions. This product also allows the user to design and document the test cases at the same time which saves an enormous amount of time and effort. Thus, in near future we will see more tools like these with even higher degree of performance than before, this will optimize the process of software testing by making it more efficient and less time consuming.