Introduction to Software Testing

Introduction

- The importance of testing in the software development life cycle
- Describe the evolution and types of software testing
- Define a bug and illustrate the reason of occurrence of a bug
- Describe the software development models

Introduction

- The success of any software product or application is greatly dependent on its quality.
- Today, testing is seen as the best way to ensure the quality of any product.
- Quality testing can greatly reduce the cascading impact of rework of projects, which have the capability of increasing the budgets and delaying the schedule.
- The need for testing is increasing, as businesses face pressure to develop sophisticated applications in shorter timeframes.

Introduction

- Testing is a method of investigation conducted to assess the quality of the software product or service. It is also the process of checking the correctness of a product and assessing how well it works.
- Testing identifies the defects in a product by following a method of comparison, where the behavior and the state of a particular product is compared against a set of standards which include specifications, contracts, and past versions of the product

Introduction

- Software testing is an incremental and iterative process to detect a mismatch, a defect or an error.

- IEEE 83a, “Software testing is the process of exercising or evaluating a system or system component by manual or automated means to verify that it satisfies specified requirements.”

Software Testing

- Software testing is an integral part of the software development life cycle which identifies the defects, flaws or the errors in the application.

Objectives of software testing:

1. It ensures if the solutions meet the business requirements, thereby enhancing customer confidence.
2. It catches the bugs, errors and defects.
3. It ensures if the system is stable and ready for use.
4. It identifies the areas of weakness in an application or product.
5. It establishes the degree of quality.
6. It determines user acceptability

Software Testing

- Testing can be comprehensive and could examine components like business requirements, design requirements, programmer's code, hardware configuration, and systems administration standards and constraints.
- Testing could also involve testing with respect to industry standards and professional best practices.
- Testing provides an opportunity to **validate** and verify all aspects of software engineering.

FACT

- In the year 2003, social security checks for 50,000 people were mailed by the U.S. Treasury Department without any beneficiary name. It was later found out that the missing names were due to a software program maintenance error.

Evolution of Software Testing

- In the early days, software was developed by small groups of people, but had to be maintained by a different set of people.
- People who were maintaining the software had very bad experiences, as the software would throw up a lot of errors.
- Due to this, it was difficult to handle large projects and thus, tasks were not completed on time and within budget.
- Resulting, project delays and some were abandoned halfway.

Evolution of Software Testing

- Software testing evolved over time. Advances in software development brought in new changes in the way testing was perceived, which led to systematic improvement in testing.

Evolution of Software Testing

- 1. ***Until 1956 - Debugging Oriented:*** Until 1956, testing and debugging were not differentiated, and testing meant debugging.
- ***1957–1978 - Demonstration Oriented:***
- During this period, the goal of testing was to make sure that the software satisfies its specifications. The period of 70s also witnessed the acceptance of the idea that software could be tested exhaustively.
- ***1979–1982 - Destruction Oriented:*** During this period, testing grew in importance with contributions from Myers (author of "The Art of Software Testing"). Test cases that had the intent of finding errors were designed and verification and validation activities were initiated.

Evolution of Software Testing

- 4. *1983–1987 - Evaluation Oriented:*
- During this period, Guidelines for Lifecycle Validation, Verification and Testing of Computer Software was created by the National Bureau of standards in 1983. This methodology enabled testing to be integrated with analysis, design and implementation of the software lifecycle.
- *1988-2000-Prevention Oriented:* During this period, the goal of testing metamorphed into a mechanism to prevent errors rather than just detecting errors. Hetzel, in 1991, gave the definition of Testing as "*Testing is planning, designing, building, maintaining and executing tests and test environments*". There was greater emphasis of early test design and a process, consisting of three phases test planning, test analysis, and test design evolved.

Software Bug

- software bug, is an error in a software program that may produce incorrect,undesired result or prevent the program from working correctly.
- In software testing, a bug not only means an error, but anything that affects the quality of the software program.
- Software bugs take different names such as – defect, fault, problem, error, incident, anomaly, failure, variance and inconsistency etc.

Conditions for a software bug

1. If the software does not respond or act in the way as stipulated in the product specification.
2. If the software behaves in a way that is stipulated in an opposite way in the product specification.
3. If the software responds or reacts in a way that is not mentioned in the product specification.
4. If the software does not behave in the mandatory way as expected --perhaps, this might not be mentioned in the product specification.
5. If the software is difficult to understand or has cumbersome steps to follow, or if it is slow in its reaction.

Example

- Let us apply the rules to a calculator.
- Assume the calculator perform the operations of addition, subtraction, multiplication and division.
- + key does not work, the bug follows the rule specified in first condition, namely, no response in the way as
- stipulated in the specification.
- The calculator is not expected to crash and freeze, and if it does, the bug follows the rule specified in condition 2, that is behaving in the opposite way as stipulated in the specification.

Example

- If there are additional features found in the calculator, namely a function to find the squares of a number, but is not mentioned in the specification, it falls in the rule specified in the third condition, where there is no mention in the product specification.
- Let us assume that the battery condition is weak, and at this stage, the calculator does not give the right answers. This condition is a typical situation which follows condition four.
- Condition 5 brings the customer perspective. It occurs whenever the performance of the software is not in an acceptable way to the user – a sample case where the tester finds the space allotted for the button very small or if the lights are too flashy that the user is unable to see the answer, it follows condition 5. The feedback of the tester becomes important, as it brings out the negatives even before the product reaches the customer.

Example

- The numerical key, zero, is clicked, the value is not displayed on the screen. This might have occurred due to many reasons- hardware or software related issues. However, though no error occurs in such a condition, the quality parameter of the mobile application is compromised, and hence it is considered as a bug.

Types of Bugs

1. Bugs due to conceptual error
   - Incorrect usage of syntax in the program, misspelled keywords, using wrong or improper design or concept.
2. Math bugs
   - Divide by zero error, overflow or underflow, lack of precision in arithmetic values due to incorrect rounding or truncation of decimal values.
3. Logical bugs
   - Infinite loops, infinite recursion, applying wrong logic, incorrect usage of jump or break conditions.

Types of Bugs

- Resource bugs
- Stack or buffer overflow, access violations, using variables that are not initialized.
- Co-programming bugs
- Concurrency errors, deadlock, race condition.
- Team working bugs
- Out of date comments, non-matching of documentation or files, linking the program to incorrect files.

Occurrence of a Bug

- Some of the common reasons that may cause bugs in software are:

Human Factor:

The developer who develops the software is human and there is always a chance that some error, incorrect usage of logic or syntax or improper linking of files might occur. Since the entire software development process involves humans, it is prone to errors.

Communication Failure:

The lack of communication or communicating incorrect information during the various phases of software development will also lead to bugs. Lack of communication can happen during various stages of Software Development Life Cycle (Requirements, Design, Development, Testing, and Modification). Many bugs occur due to lack of communication when a developer tries to modify software developed by another developer.

Occurrence of a Bug

- Unrealistic Development Timeframe:

- Sometimes developers work under tight schedules, with limited or insufficient resources, and unrealistic project deadlines. Compromises are made in the requirements or design of the software in order to meet the delivery requirements. Most of the time, it leads to wrong understanding of the requirements, inefficient design, improper development, and inadequate testing.

Occurrence of a Bug

- Poor Design Logic:

During complex software development, some amount of R&D and brainstorming has to be done to develop a reliable design. It is also important to choose the right components, products and techniques for software development and testing during the design phase. If there is any lack of proper understanding in the technical feasibility of the components, products, or techniques in the development process, it may cause bugs in the software that is being developed.

Occurrence of a Bug

- Poor Coding Practices:

- Poor coding practices such as inefficient use of codes, misspelled keywords, or incorrect validation of variables will lead to bugs. Sometimes, faulty tools such as editors, compilers, and debuggers generate wrong codes, which cause errors in the software and it is difficult to debug such errors.

Occurrence of a Bug

- Lack of Skilled Testing:

If there are any drawbacks in the testing process of the software, bugs go unnoticed.

- Change Requests:

If there are last minute changes on requirements, tools, or platform, it could cause errors and can lead to serious problems in the application.

Cost of Bugs

- Any bug that exists in the software delivered to the customer can lead to huge financial losses. It will bring down the reputation of the organization developing the software. In case the bug causes any

- Damage to life or property of the customer, it can also lead to huge penalty on the organization that developed the software. Bugs that are detected during the testing stage will take some time to get fixed, affecting the timeline and increasing the project cost.

Cost of Bugs

- Bugs detected and fixed at the earliest in the software development life cycle, it can result in higher return on investment for the organization with respect to time and cost.
- The cost of fixing a bug differs depending on the development stage at which it is detected.

Cost of Bugs

1. Requirement Stage:

- It would cost the organization some time to rewrite the requirements and can be easily detected and fixed.

- Coding Stage:
- The developer will spend some time debugging. However, the time required to fix the bug will vary depending on the complexity of the bug. A bug detected by the developer at this stage can be resolved easily, since the developer understands the problem and will be able to fix it.

Cost of Bugs

- Integration Stage:
- It will require more time to be resolved. Since the problem occurs at a higher level, it requires time to check the part of the code or configuration which is wrong and additional time to fix the bug. The developer and other system engineers will be involved in detecting and resolving the bug.

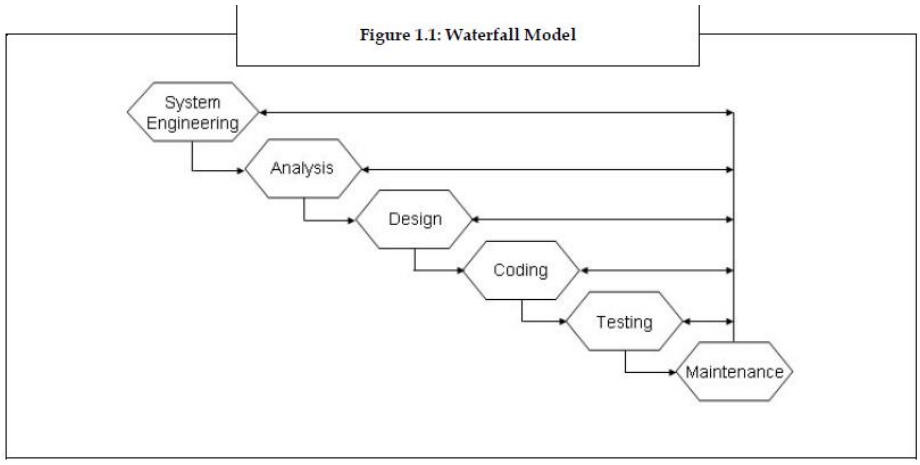
- Testing Stage:
- It will involve the developer, system engineer, and project manager for detecting and resolving the bug. It is an iterative process, which takes a lot of time and effort with respect to man hours and cost in order to detect and fix the bug. Since the bugs have to be tracked and prioritized for debugging, it will increase the project cost in terms of man power and tools required and causes delay in the delivery times as well.

Cost of Bugs

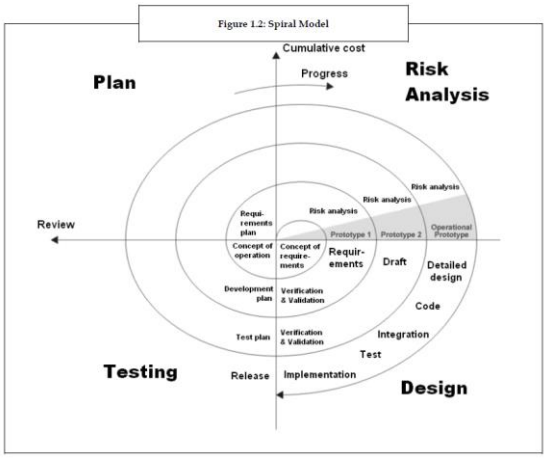
- Production Stage:
- It will take huge time and investment for the organization, as it involves developers, system engineers, project managers, and customers for detecting and resolving the bug. It demands prioritizing and detailed planning when compared to a bug detected in testing stage. Such bugs not only cause a huge loss, but also bring down the reputation of the organization.

- A bug found and fixed during the early stages – say, while defining the specification or requirements, might cost the company a very small amount, for example, consider it to be one US dollar per bug. If the same bug is found during the coding and testing stage, it costs the company reasonably, for example, 10 to 100 US dollars. If the same bug is found by the customer the cost will be very high a few thousand dollars along with the reputation of the company.

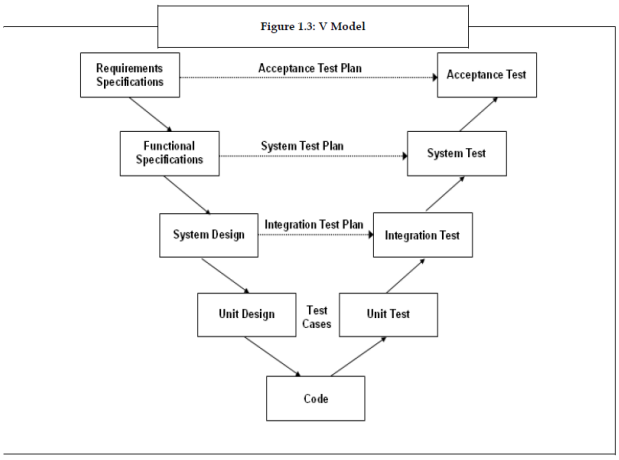
Waterfall Model – Software Development



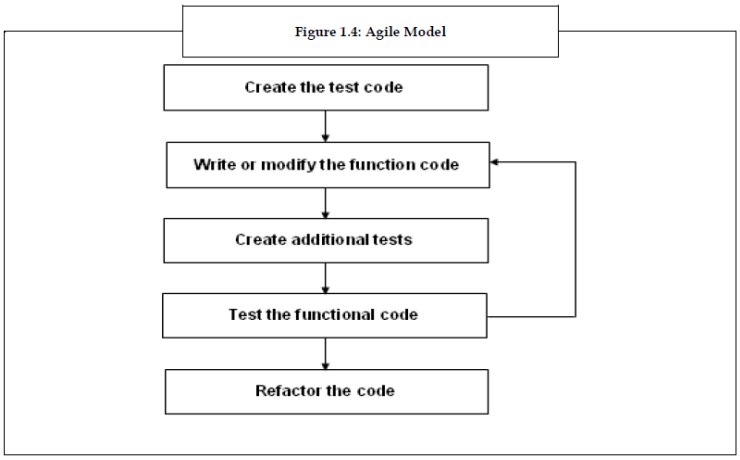
Spiral Model – Software Development



V model – Software Development



Agile Model – Software Development



Questions ?