SOFTWARE TESTING

To err is human

Since committing errors is an inevitable component in all facets of any application development, it is highly required to detect and eliminate the errors. In software projects, error injection is a common act which can get seeded either in the product or during the process of developing the product. Hence, such errors should be identified and eliminated. There is always difference between error, fault, failure and defect.

IEEE definitions for the above are as below:

Error: Human mistake that caused fault
Fault: Discrepancy in code that causes a failure.
Failure: External behavior is incorrect
Defect: Fault when become visible can act as a defect.

There are two ways in which errors can be addressed namely quality control and quality assurance.

Quality Control – The process by which product quality is compared and detected with respect to requirements and other relevant specifications, focus is in detection and removal. Testing is one activity through which quality control can be achieved.

Quality Assurance – The set of activities (including facilitation, training, measurement, and analysis) needed to provide adequate confidence that process are established continuously improved to produce products that meet specifications and are fit for use. There are several techniques through which quality assurance can be achieved namely reviews, walkthroughs, inspection, training, audit assessment, metrics and standards through which process can be improved.

Software Testing: A Quality control activity aimed at evaluating a software item against the given system requirements

Some definitions of testing:

- Definition (1)
  - Process of exercising or evaluating a system or system component by manual or automated means to verify that it satisfies specified requirement (IEEE 83a)
- Definition (2)
  - Structured process that uncovers the defects in a software product.
  - Destructive in nature (dismantling the wishful assumption that code is bug-free)
- Definition (3)
  - Testing is a process of executing a program with the intent of finding errors (Myers).

Significance of testing:

- Ensure product works with negligible risks.
- Find defects in product
- Demonstrate the lack of quality
- Client or End user should not find bugs
- Detect programming errors
- Demonstrate difference between specifications and developed system
- Establish confidence in the product

Limitations of testing:

- Does not generalize system behavior
- Does not guarantee bug free product
- No substitute for good programming

Testing can be classified into various types of testing and various levels of testing

Levels of Testing:

- Unit test
- Integration test
- System test
- User acceptance test

Unit testing is an activity which will be conducted by both testers and developers. It is act where every deliverable which is developed by the developer will be tested and also testers will test it from both functionality check and non functionality checks. Example: Testing the login module of online library management system.

Integration testing is also usually tested by programmers. However, there can be testers who do interface testing. Main issue in this type of testing is the choice of integrating i.e top down integration test or bottom up integration test or sandwich integration test. Example: Testing the login module with book taken module and payment module of the online library management system.

System test is an activity where the developed system after unit tested, integration tested will be once again test the entire system in order to validate if all specifications are met or not. This will be usually tested by testers. Example: Testing complete online library management system.

User acceptance test is the final test subjected to the customers to test the product before it is deployed to them. This test decides to either accept the product or not depending on satisfaction level of all specifications in the product developed. Example: Testing the online library management system by customer.

Types of testing:

- Alpha test
- Beta test
- Black box test
- While box test
- Random test
- Regression test
- Performance test
- Load test
- Stress test
- Exploratory test
- Combinatorial test
- Smoke test

i. Alpha test:

Testing the product by the developers in presence of customers in house of development is alpha test. Example: Testing online library management software in the company who developed and showing them how it operates to customers

ii. Beta test:

Testing the product by customers at their location in absence of development team is beta testing. Example: Testing online library management software in the customer's location by them.

iii. Black box testing:

Testing the product by giving input and obtaining the output is black box test. Example: Giving login and checking if it logs in or not in the online library management system

iv. White box test:

Testing the code either line by line or condition by condition or loop by loop or by path are white box test. Example: tracing every statement of login module of online library management system.

v. Random test:

Random testing involves selecting test cases based on a probability distribution. It is not ad hoc test. Example: Testing the online library management system by giving random number of customers to login and check if they are legitimate users or not

vi. Regression test:

This is the most common test run on any application. Intention is not just to detect defects in the product but also once debugged, the corrected module is once again subjected to test from the beginning and also to test all those modules to which the corrected module has a dependency.

Example: test login module and found a error while navigating to menu page. Having debugged the error, once again retest login module and also all those modules to which this login is associated with. Aim of such test is to check for bad fixes. Bad fix is introducing new defects by debugging exiting defect.

vii. Performance test:

Performance testing is the process of determining the system's performance that includes speed, reliability under varying load. Example: Test the login module by readers to login in large numbers to check if the system goes to down time or does performance decrease. In this case, it is considered to test the login module in normal conditions.

viii. Load test:

Load testing is the process of determination of behavior of system when multiple users access it at the same time. This is once again testing with expected load given to system. Example: test the behavior of login module when 1000 readers login at the same time.

ix. Stress test:

Stress Testing is performed to test the robustness of the system or software application under extreme load. Example: Making 1500 readers to login at the same time and to test to see if the performance is decreased or not.

x. Exploratory test:

Exploratory testing is one such manual testing approach where test cases are generated based on the experience and knowledge of the tester. Test cases in exploratory testing are not pre defined unlike conventional testing but are generated and executed simultaneously. Example: Testers testing login module with already generated test case find possibility of testing with few more creative test cases which are not available in test suite.

xi. Combinatorial test:

Combinatorial testing (CT) which is a black box testing method is carried out on how the variable and their combinations are selected to perform proficient testing with the least possible number of test cases covering maximum critical aspects of the software. Example: Testing online library management system with all possible combinations of inputs and parameters for specific scenarios.

xii. Smoke test:

Testing the critical functionalities of the system is smoke test. Example: Testing payment dues module of online library management system.

Software testing however a life cycle to be has followed which is popularly known as Software Test Life Cycle (STLC).

Once STLC begins, following activities are conducted as part of test life cycle:

- Test Strategy
- Test Plan
- Test Case Design
- Test Case Generation
- Test Execution
- Regression Test
- Test Closure Report
- Retrospection Move

Test Strategy: This activity decides the approach of testing which includes decision on type of testing, rationale for it so on.

Test Plan: This activity ensures one to plan for the strategy incorporated in terms of time, cost, resources required.

Test Case Design: In this activity, testers who are assigned comes out with design of test case which looks as below

Table 6: Sample Test Case template

Test id	Test case description	Steps followed	Input Data	Expected Output	Actual Output	Test case passed/failed	Remarks

Test Case Generation: In this activity, test case is generated where every specification is analyzed from all perfectives of stakeholders and prepared. Collection of test cases is test suite.

Table 7: Sample Test Case Generation for ATM application as an example

Test id	Test case description	Steps followed	Input Data	Expected Output	Actual Output	Test case passed/failed	Remarks
R101	Login	1	Insert Card	Card is valid			
		2	Enter 3 digit numeric PIN	PIN Is valid			

Test Execution: In this activity, testers carry out actual testing where code will be subjected to test for the test cases generated.

Table 8: Test Execution for the ATM Application as an example for login module sample

Test id	Test case description	Steps followed	Input Data	Expected Output	Actual Output	Test case passed/failed	Remarks
R101	Login	1	Insert Card	Card is valid	Valid Card	Passed	
		2	Enter 3 digit numeric PIN	PIN Is valid	PIN Invalid	Failed	Accepted alphanumeric characters as PIN

Regression Test: This is the most common activity that occurs while testing any application of any domain. During this part of STLC, testers once they identify the defect, it will be sent to the authors of the code to rectify the defect. Having obtained the debugged code, the code will be tested from the beginning and also tested for bad fixes in the modules to which this code has a dependency.

Test Closure Report: In this activity, testers will prepare a report on how many defects were identified, severity and impact of each of the defect, time taken to identify the defect, time taken to rectify and resolve the defect by developers, number of bad fixes reported, test effort and efficiency achieved. The report will be kept in the repository for subsequent use and knowledge.

Retrospection Move: Once the testing process is completed and test report is handed over to the project manager or quality head, testers will introspect to analyze the strengths and weakness of the entire STLC process to find out what went well and what did not go well.

Verification and Validation:

Above process of STLC when takes place, the testers carry out two important modes of testing namely verification and validation

Verification:

"Are we building the product right". It is subjective in nature and conducted for every deliverable. Hence, the software should conform to its specification.

Validation:

"Are we building the right product". It is objective in nature and is conducted for the end product or artefacts. Hence, the software should do what the user really requires.

Software Inspection

Software inspection is visual examination of software to find out the static defects. Static defects are the ones which can be detected through our eyes than requiring running the system. It is usually carried out by an external inspector who is not part of the project and can be either from other project within the organization or an external person to the organization. Carrying out inspection ensures reduction of test time as most of the static defects will be detected through inspection and testing will have to detect only the dynamic defects which are visible only at the execution time.

Software inspection can be carried out for every deliverable such as given below

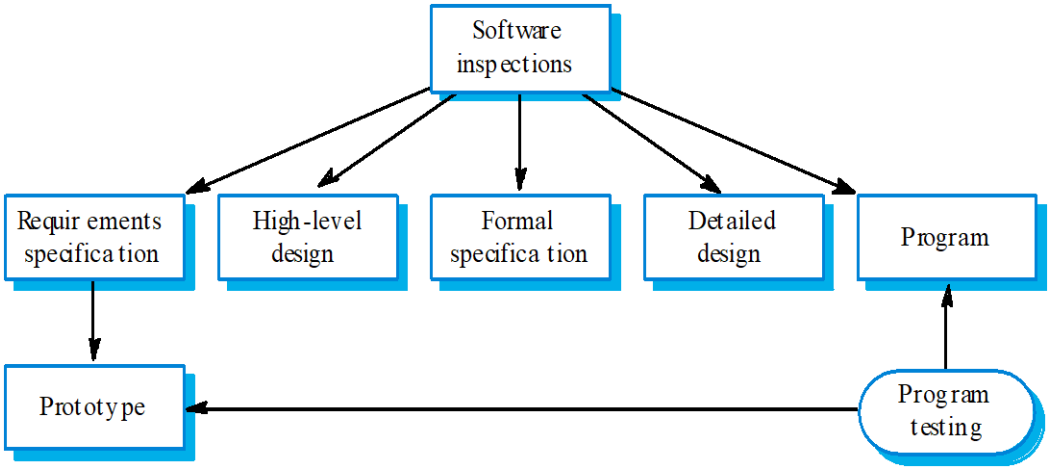

Figure 37: Deliverable upon which software inspection can be conducted.

From the figure, it is clear that software inspection can be carried out on every artifact of every phase of software development.

However, inspection cannot just happen without prior plan and domain knowledge. The figure below shows the preparation phase required to carry out inspection.

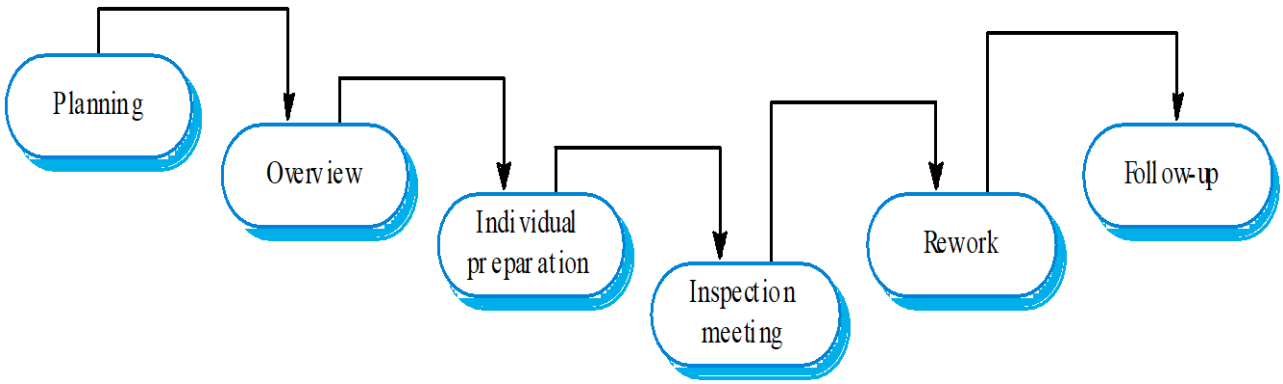

Figure 38: Inspection Process

From the figure, it is clear that inspection will be initially planned for the choice of inspector and their team, time to be given for them, resources to be supported for the inspection process and environment to be set up for the same.

With this plan, an overview of the agenda of the inspection will be prepared to check which modules and which of the deliverables needs to be software inspected. Subsequently, a small amount of time will be provided for the team to prepare themselves for carrying out inspection so that they are aware of the dolman, and other necessary knowledge to do inspection. Actual inspection occurs where defects will be identified and reported. Finally, when defects are fixed and given again for re inspection, if any defects are identified, it will be given for resolving. This process should not be carried out more than two times to ensure effective inspection process.

Thus, the inspection team has their defined roles and responsibilities while doing inspection. This is summarized in the table below

Table 9: Roles and responsibilities of the inspection team members

Author or owner	The programmer or designer responsible for producing the program or document. Responsible for fixing defects discovered during the inspection process.
Inspector	Finds errors, omissions and inconsistencies in programs and documents. May also identify broader issues that are outside the scope of the inspection team.
Reader	Presents the code or document at an inspection meeting.
Scribe	Records the results of the inspection meeting.
Chairman or moderator	Manages the process and facilitates the inspection. Reports process results to the Chief moderator.
Chief moderator	Responsible for inspection process improvements, checklist updating, standards development etc.

Table above indicates how roles are defined for every member of the inspection team. They will be provided with checklist through which they can verify if the data is defined, if parameters are properly assigned and so on. Deviation of the deliverable from the checklist indicates existence of defects.

While planning the testing, usually V Model is followed as shown below

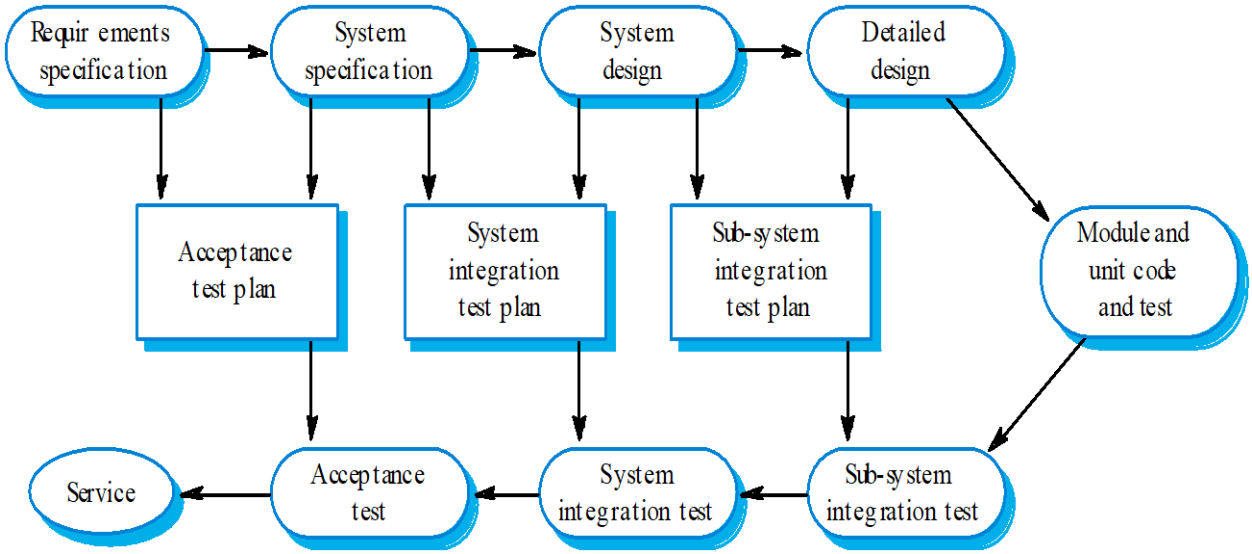

Figure 39: V Model of carrying out Testing

Figure indicates that code is unit tested by developers and testers, once unit code is tested, testers carry out integration testing by integrating the modules and test the subsystem thereby. This mode of testing checks to test the design if design prepared and integrated modules are both matching or has any discrepancy. With sub system testing which tests the design and integration modules along with their interfaces and parameters passing between modules, next step is it to test the entire system. Testers then test system to check if all specifications are met or not. This type of testing is validation testing to see the code maps to specifications. Finally, user acceptance test is conducted to validate if users are satisfied with the system which is developed and if all their requirements are well achieved.

Yet another way of performing testing for critical applications such as software for robotic surgery, space craft to be launched in the space and applications of similar types are tested using clean room approach.

The name is derived from the 'Cleanroom' process in semiconductor fabrication. The philosophy is defect avoidance rather than defect removal. This approach follows meticulous mode of testing to ensure that the product is clean from all perspectives. This software development process is based on Incremental development, Formal specification, Static verification using correctness arguments and Statistical testing to determine program reliability. Figure below indicates the process of cleanroom testing

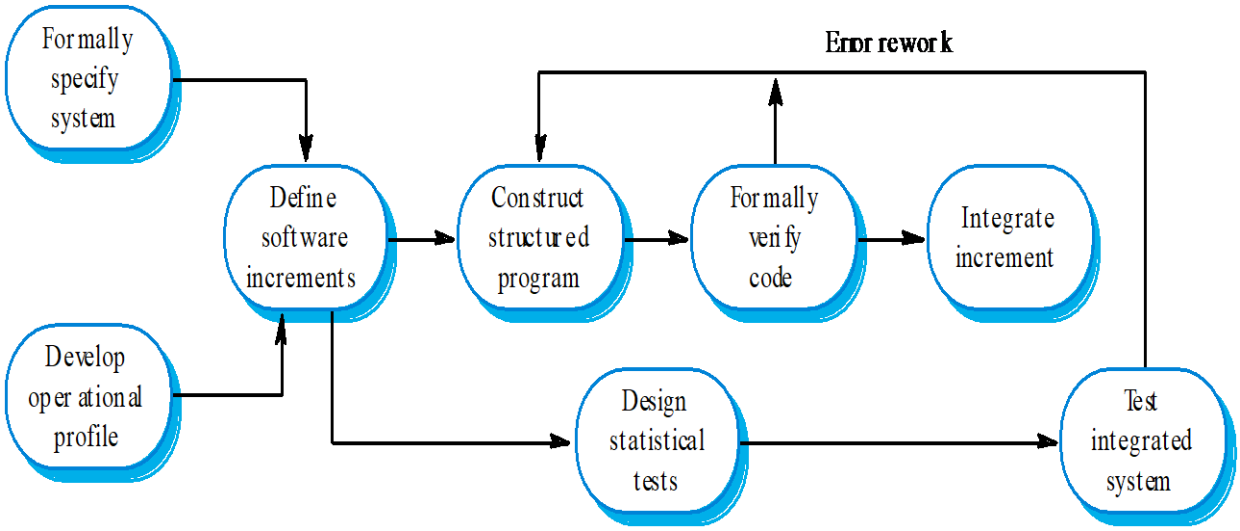

Figure 40: Cleanroom approach of testing

In this approach, requirements are specified in a formal way where mathematically specifications have to be expressed and written. Operational profile in testing indicates a quantitative way of indicating how a system is used by writing the inputs (profile) and anticipating the type of outputs accordingly. With the inputs for the formally written down specifications, every increment of the application will be identified and programs will be written to implement the same. The implemented code will be verified unit wise formally, integrate them and test it in a statistical way using statistical testing such as reliability growth models or any other tests. The final system if satisfied will be then delivered to the customers. This mode of testing in a formal way using statistical testing approach for the formally specified requirements ensures maximum defect free products.

This approach is carried out by a set of team members who include:

- Specification team: who are responsible for developing and maintaining the system specifications
- Development team: who is responsible for developing and verifying the software and the software is NOT executed or even compiled during this process
- Certification team: who are responsible for developing a set of statistical tests to exercise the software after development and generally Reliability growth models are used to determine when reliability is acceptable.