



Software Verification and Validation

Software Verification and Validation

➤ **Software Verification**

- Verification is the process of confirming if the software is in accordance to the business requirements, and is developed adhering to the proper specifications and methodologies.
 - Verification ensures the product being developed is according to design specifications.
 - Verification answers the question– "Are we developing this product by firmly following all design specifications?"
 - Verifications concentrates on the design and system specifications.

Software Verification and Validation

➤ **Software Validation**

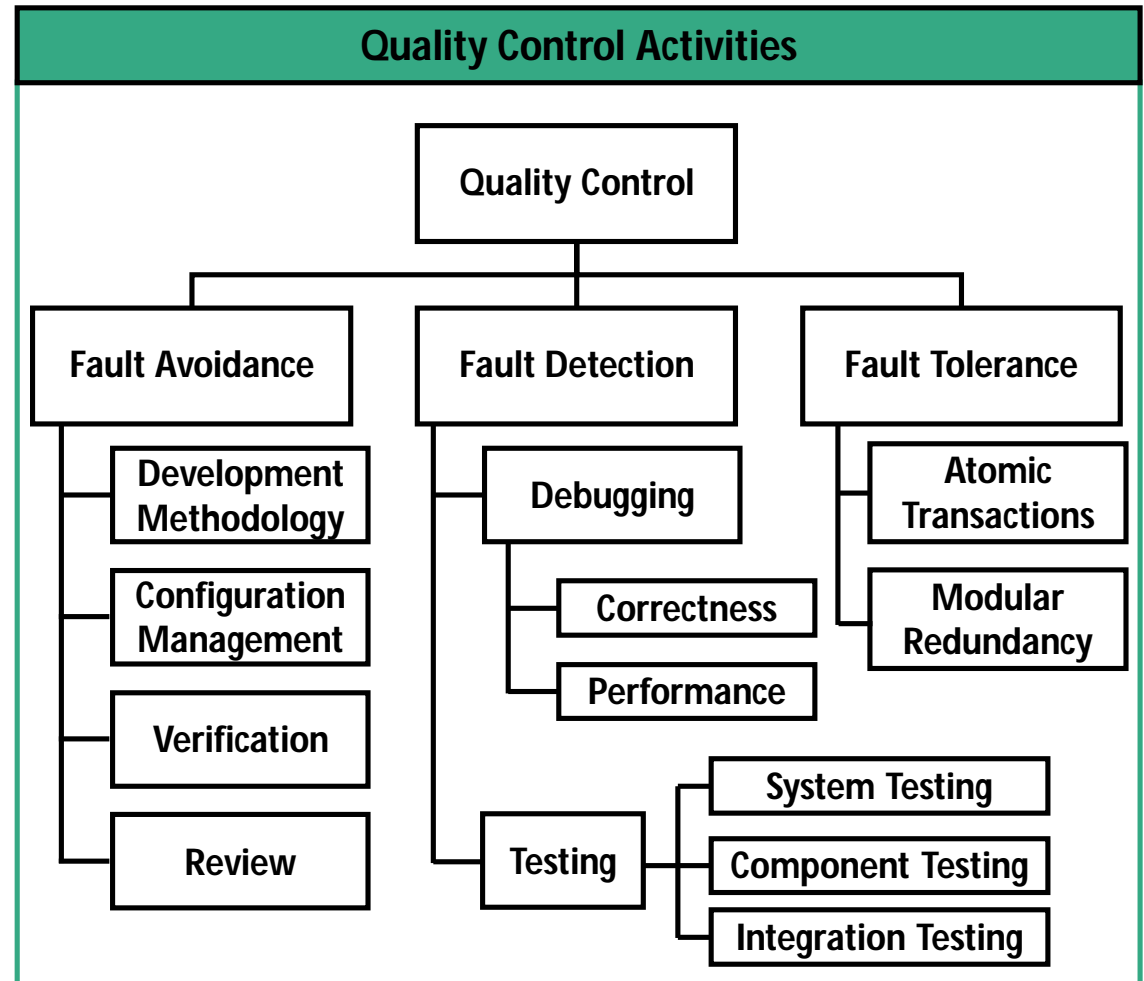
- Validation is process of examining whether or not the software satisfies the user requirements. It is carried out at the end of the SDLC. If the software matches requirements for which it was made, it is validated.
 - Validation ensures the product under development is as per the user requirements.
 - Validation answers the question - "Are we developing the product which attempts all that user needs from this software ?".
 - Validation emphasizes on user requirements.


Software Verification and Validation

➤ Target of the test are -

- **Errors** - These are actual coding mistakes made by developers. In addition, there is a difference in output of software and desired output, is considered as an error.
- **Fault** - When error exists fault occurs. A fault, also known as a bug, is a result of an error which can cause system to fail.
- **Failure** - failure is said to be the inability of the system to perform the desired task. Failure occurs when fault exists in the system.

Software Verification and Validation





Manual Vs. Automated Testing

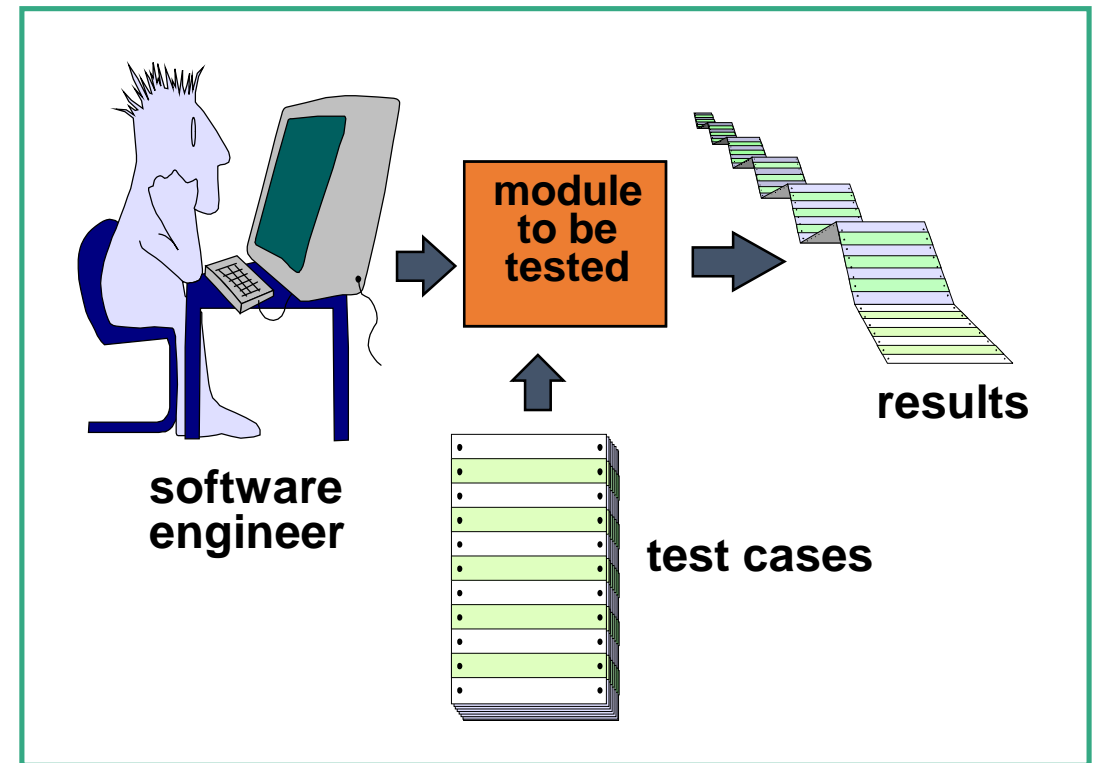
Manual Vs. Automated Testing

- Testing can either be done manually or using an automated testing tool:
 - **Manual** - This testing is performed without taking help of automated testing tools. The software tester prepares test cases for different sections and levels of the code, executes the tests and reports the result to the manager.
 - Manual testing is time and resource consuming. The tester needs to confirm whether or not right test cases are used. Major portion of testing involves manual testing.
 - **Automated** - This testing is a testing procedure done with aid of automated testing tools. The limitations with manual testing can be overcome using automated test tools.

Manual Vs. Automated Testing

- A test needs to check if a webpage can be opened in Internet Explorer. This can be easily done with manual testing. But to check if the web-server can take the load of 1 million users, it is quite impossible to test manually.
- There are software and hardware tools which helps tester in conducting load testing, stress testing, regression testing.

Manual Vs. Automated Testing





Testing Approaches

Testing Approaches

➤ Tests can be conducted based on two approaches –

1. Functionality testing

2. Implementation testing

When functionality is being tested without taking the actual implementation in concern it is known as black-box testing. The other side is known as white-box testing where not only functionality is tested but the way it is implemented is also analyzed.

Exhaustive tests are the best-desired method for a perfect testing. Every single possible value in the range of the input and output values is tested. It is not possible to test each and every value in real world scenario if the range of values is large.

Testing Approaches



Developer

Understands the system
but, will test “gently” and,
is driven by “delivery”



Independent Tester

Must learn about the system,
will attempt to break it
and, is driven by quality



Black Box Testing

Black Box Testing

It is carried out to test functionality of the program and also called 'Behavioral' testing. The tester in this case, has a set of input values and respective desired results. On providing input, if the output matches with the desired results, the program is tested 'ok', and problematic otherwise.



Black Box Testing

- In this testing method, the design and structure of the code are not known to the tester, and testing engineers and end users conduct this test on the software.
- **Black-box testing techniques:**
 - **Equivalence class** - The input is divided into similar classes. If one element of a class passes the test, it is assumed that all the class is passed.
 - **Boundary values** - The input is divided into higher and lower end values. If these values pass the test, it is assumed that all values in between may pass too.

Black Box Testing

- **Cause-effect graphing** - In both previous methods, only one input value at a time is tested. Cause (input) – Effect (output) is a testing technique where combinations of input values are tested in a systematic way.
- **Pair-wise Testing** - The behaviour of software depends on multiple parameters. In pairwise testing, the multiple parameters are tested pairwise for their different values.
- **State-based testing** - The system changes state on provision of input. These systems are tested based on their states and input.



White Box Testing

White Box Testing

- It is conducted to test program and its implementation, in order to improve code efficiency or structure. It is also known as 'Structural' testing.
- In this testing method, the design and structure of the code are known to the tester. Programmers of the code conduct this test on the code.



White Box Testing

- The below are some White-box testing techniques:
 - **Control-flow testing** - The purpose of the control-flow testing to set up test cases which covers all statements and branch conditions. The branch conditions are tested for both being true and false, so that all statements can be covered.
 - **Data-flow testing** - This testing technique emphasis to cover all the data variables included in the program. It tests where the variables were declared and defined and where they were used or changed.



Testing Levels

Testing Levels

- Testing itself may be defined at various levels of SDLC. The testing process runs parallel to software development. Before jumping on the next stage, a stage is tested, validated and verified.
- Testing separately is done just to make sure that there are no hidden bugs or issues left in the software. Software is tested on various levels -
 - **Unit Testing**
 - **Integration Testing**
 - **System Testing**
 - ✓ **Functionality testing**
 - ✓ **Performance testing.**
 - ✓ **Security & Portability**
 - **Acceptance Testing**
 - ✓ **Alpha testing**
 - ✓ **Beta testing**
 - **Regression Testing**

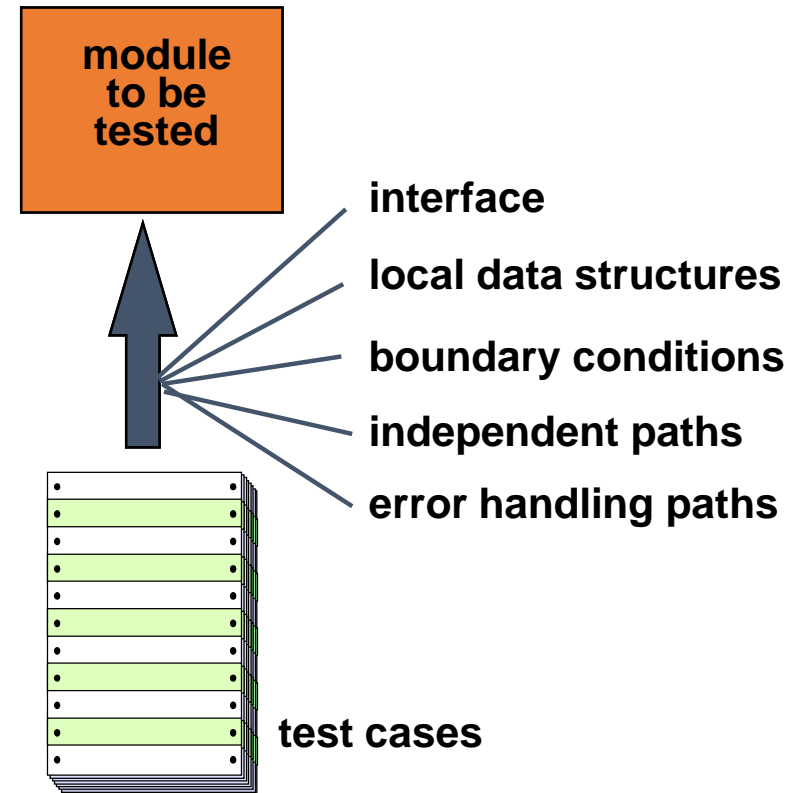
Testing Levels

➤ Unit Testing

- While coding, the programmer performs some tests on that unit of program to know if it is error free. Testing is performed under white-box testing approach. Unit testing helps developers decide that individual units of the program are working as per requirement and are error free.

Testing Levels

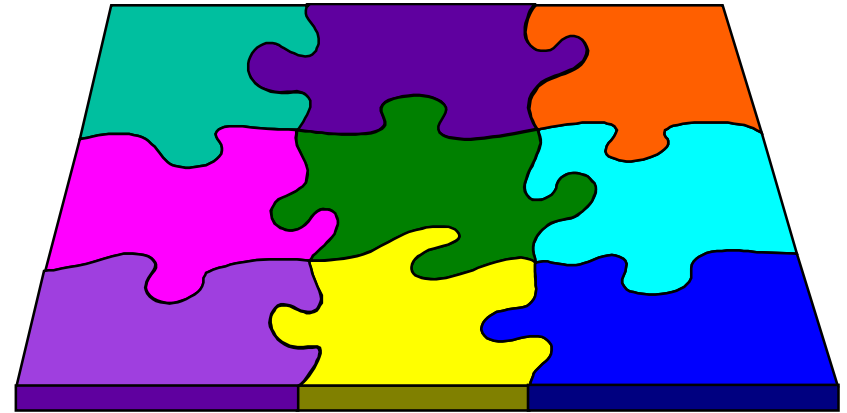
➤ Unit Testing



Testing Levels

➤ Integration Testing

- Even if the units of software are working fine individually, there is a need to find out if the units if integrated together would also work without errors. For example, argument passing and data updation etc.



Testing Levels

➤ System Testing

- The software is compiled as product and then it is tested as a whole. This can be accomplished using one or more of the following tests:
 - **Functionality testing** - Tests all functionalities of the software against the requirement.
 - **Performance testing** - This test proves how efficient the software is. It tests the effectiveness and average time taken by the software to do desired task. Performance testing is done by means of load testing and stress testing where the software is put under high user and data load under various environment conditions.
 - **Security & Portability** - These tests are done when the software is meant to work on various platforms and accessed by number of persons.

Testing Levels

➤ **Acceptance Testing**

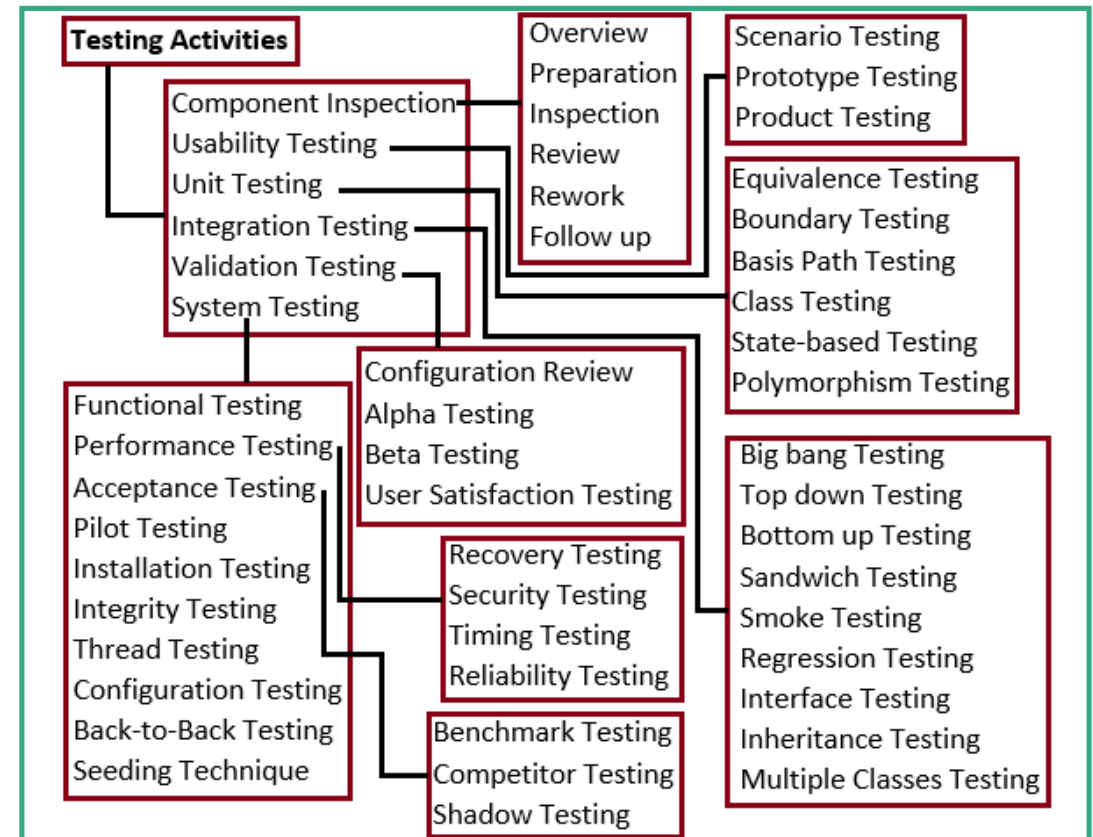
- When the software is ready to hand over to the customer it has to go through last phase of testing where it is tested for user-interaction and response. This is important because even if the software matches all user requirements and if user does not like the way it appears or works, it may be rejected.
- **Alpha testing** - The team of developer themselves perform alpha testing by using the system as if it is being used in work environment. They try to find out how user would react to some action in software and how the system should respond to inputs.
- **Beta testing** - After the software is tested internally, it is handed over to the users to use it under their production environment only for testing purpose. This is not as yet the delivered product. Developers expect that users at this stage will bring minute problems, which were skipped to attend.

Testing Levels

➤ Regression Testing

- Whenever a software product is updated with new code, feature or functionality, it is tested thoroughly to detect if there is any negative impact of the added code. This is known as regression testing.

Testing Levels





Testing Documentation

Testing Documentation

- Testing documents are prepared at different stages -
 - **Before Testing**
 - **While Being Tested**
 - **After Testing**

Testing Documentation

➤ Before Testing

- Testing starts with test cases generation. Following documents are needed for reference –
 - **SRS document** - Functional Requirements document
 - **Test Policy document** - This describes how far testing should take place before releasing the product.
 - **Test Strategy document** - This mentions detail aspects of test team, responsibility matrix and rights/responsibility of test manager and test engineer.

Testing Documentation

➤ Before Testing (Contd.)

- **Traceability Matrix document** - This is SDLC document, which is related to requirement gathering process. As new requirements come, they are added to this matrix. These matrices help testers know the source of requirement. They can be traced forward and backward.

Testing Documentation

➤ While Being Tested

- The following documents may be required while testing is started and is being done:
 - **Test Case document** - This document contains list of tests required to be conducted. It includes Unit test plan, Integration test plan, System test plan and Acceptance test plan.
 - **Test description** - This document is a detailed description of all test cases and procedures to execute them.
 - **Test case report** - This document contains test case report as a result of the test.
 - **Test logs** - This document contains test logs for every test case report.

Testing Documentation

➤ After Testing

- The following documents may be generated after testing :
- **Test summary** - This test summary is collective analysis of all test reports and logs. It summarizes and concludes if the software is ready to be launched. The software is released under version control system if it is ready to launch.