- Validating Testing

Validation testing begins after integration testing.

Focused on actions which are directly visible by the user.

Validation is successful when the functionality of software is as per expectations of the customer.

Main concern is user requirements.

Validation test criteria

- All of the expected functional requirements are fulfilled.
- Behavioral characteristics are achieved
- Accuracy in content and properly presented
- Consideration of all performance requirements
- Accurate documentation.

Types of validation testing

1. Alpha Testing
2. Beta Testing

Alpha Testing

The testing is done to find out bugs before deploying the software application to the end user.

Objective is to find and fix bugs that were not discovered through previous tests.

Performed by in-house software engineers or QA staff.

It is the final testing stage before the software is released into the real world.

Performed in 2 phases.

- In the first phase, software is tested by development team members. They perform debugging of software to catch bugs quickly.

- In the second phase, software is tested by the software quality analyst team for additional testing in the actual user's environment setup.

Advantages

- Better insight about software reliability at its early stages.
- Reduce delivery time, free up team for other projects.
- Early feedback helps to improve software quality.

Beta Testing

Performed at the location of customer.

Actual as well as intended users will test the software to determine whether the software is satisfying their expectations.

Allows users to test software before it is released to the public.

Minimizes the product failure risks.

Ensures reliability, security, robustness etc. from the user's perspective.

Type of Beta testing : Traditional, public, technical, focused and past release.

Beta testing is also called as User acceptance testing, customer acceptance testing, customer validation testing or pre-release testing

Advantages

- Decreases product failure risks.
- Improves software quality by using feedback from customers.
- Cost effective as compared to other data collection techniques.
- Improves customer satisfaction.

Types of Beta Testing

1. Traditional Beta testing

Software product is provided to the targeted end user and associated data is collected. This data is useful for product improvement.

2. Public beta testing

Products are released publicly in the real world using online channels and data can be collected from anyone. Feedback is used for product improvements.

3. Technical Beta testing

Software products are released in the internal group of an organization and collect data from the employees of the organization.

4. Focused Beta

Product is released in the market and data is collected about specific features of the program.

5. Post release Beta

Product is released in the market and data is collected to improve the software product for the next release.

Alpha testing	Beta testing
Done by testers	Done by End users who are not employees of organization
Testing is done in lab environment	Testing is done in real world environment
Includes white box and black box testing	Includes Black box testing
Requires lot of time to perform	Few weeks of time is required for performing beta testing
Security testing and reliability are not tested in depth during alpha testing	Reliability, Security, Robustness is verified during beta testing
Important issues can be corrected by developers without delay.	Issues detected in beta testing will be corrected in future versions of the product.
Gives assurance about quality of the software product before going to beta testing	Gives assurance that the software product is ready for use in a real world environment.

- ## System Testing- Art of Debugging

Testing of entire and completely integrated software.

System testing is end to end testing. i.e. test system from login module to log out module.

Contains both functional as well as Non-functional testing.

System testing is included in black box testing.

Few Types of System Testing

Usability testing

Here, a group of end users of the software use the product to check user friendliness.

It is non-functional testing.

Test the ease of handling the application.

Also known as User Experience Testing.

Load Testing

A type of Performance Testing under real life load conditions.

Checks how the application behaves when multiple users access it at same time.

It is non-functional testing.

Normally used when we test Client/Server based applications and Web based applications.

Used to find out:

- Maximum number of users who can access it at the same time.
- Specifies whether currently available infrastructure (software and hardware) is adequate to execute the application.
- Checks what happens when the maximum number of users accesses the system simultaneously.

- Checks Scalability (increase capacity) to permit more users.

Regression Testing

Used to check whether changes made in code due to some error or change in requirement affects existing working functionality.

Here, we perform already executed test cases to give assurance that old functionalities work well after performing changes in code.

To give assurance that new code added in software does not disturb existing functionalities.

Recovery Testing

Done to specify whether the system recovers itself after the system crash due to disaster such as power or network failure.

Check whether the system will perform rollback.

Migration Testing

To give assurance that software can be moved from older system infrastructure to current system infrastructure without any problem.

Functional testing

Checks that every function present in the software application works as per requirements of the user.

Includes black box testing and it does not focus on the source code of the software.

Functionality is verified by the tester using appropriate test data and actual result is compared with expected result.

Done using Requirement Specification document.

Hardware/Software Testing

Perform testing of communication between the hardware and software used in the system.

Security testing

Verifies that protection mechanisms built into a system will, in fact, protect it from improper access.

Stress testing

Executes a system in a manner that demands resources in abnormal quantity, frequency, or volume.

Performance testing

Tests the run-time performance of software within the context of an integrated system.

Often coupled with stress testing and usually requires both hardware and software instrumentation.

Can uncover situations that lead to degradation and possible system failure.

White Box Testing

In this testing process, we verify internal coding and infrastructure of software products under the test.

Programming knowledge or detailed functional knowledge is a prerequisite for testers.

Also called Clear-Box testing, Open box testing, Structural testing, Transparent Box testing, Code-based testing, Glass Box testing.

It uses following methods:

Statement Coverage: Testing all programming statements using minimum number of tests.

Branch Coverage: to ensure that all branch conditions in the system are tested at least once.

Path Coverage: to ensure that each statement and branch in the system is tested at least once.

By using Statement and Branch coverage, we can perform 80-90% code coverage.

Other coverage types are Condition coverage, Multiple condition coverage, Path coverage, function coverage etc.

White box testing involves two steps:

Step 1: Understand the source code

Testers understand the source code of the product.

Tester must have good programming knowledge. They should be able to detect security problems and protect the software from hackers and naïve users who may add malicious code in software products.

Step 2: Create test cases and execute

This includes testing the source code of software products under the test for checking the proper flow of control and structure.

Testers will generate test cases for every process or group of processes in the product.

May be performed by developers.

Advantages:

White box testing is thorough as the entire code and structures are tested.

It results in the optimization of code removing errors and helps in removing extra lines of code.

It can start at an earlier stage as it doesn't require any interface as in the case of black box testing.

Easy to automate.

White box testing can be easily started in the Software Development Life Cycle.

Easy Code Optimization

Testers can identify defects that cannot be detected through other testing techniques.

Testers can create more comprehensive and effective test cases that cover all code paths

Disadvantages:

Testers need to have programming knowledge and access to the source code to perform tests.

Testers may focus too much on the internal workings of the software and may miss external issues.

Testers may have a biased view of the software since they are familiar with its internal workings.

1. Basis Path Testing

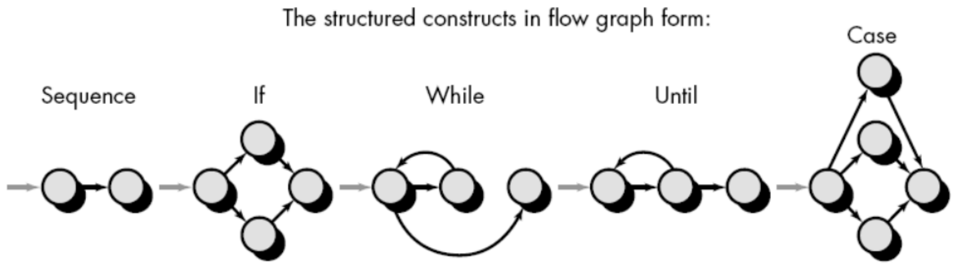
Basis path testing is a white-box testing technique first proposed by Tom McCabe

These tests guarantee to execute every statement in the program at least once during testing. Basis set is the set of all execution paths of a procedure.

Flow Graph notation

Before the basis path method can be introduced, a simple notation for the representation of control flow, called a flow graph (or program graph) must be introduced.

Flow graph depicts control flow and uses different constructs. These individual constructs are combined together to produce the flow graph for a particular procedure.



Flow Graph terminology

Node: Each circle, called a flow graph node, represents one or more procedural statements. A sequence of process boxes and a decision diamond can map into a single node. Each node that contains a condition is called a predicate node.

Edge: Edge is the connection between two nodes. It represents flow of control and is analogous to flowchart arrows. An edge must terminate at a node, even if the node does not represent any procedural statements.

Region: Areas bounded by edges and nodes are called regions.

Cyclomatic Complexity

Cyclomatic complexity is a software metric that provides a quantitative measure of the logical complexity of a program.

The value computed for cyclomatic complexity defines the number of independent paths.

Independent path is an execution flow from the start point to the end point. There can be various execution paths depending upon the decision taken on the control statement.

It provides us with an upper bound for the number of tests that must be conducted, because for each independent path, a test should be conducted to see if it actually reaches the end point or not.

Deriving the test cases

The main objective of basic path testing is to derive the test cases for the procedure under test. The process of deriving test cases is as follows:

From source code, derive flow graphs.

Determine the Cyclomatic complexity, V(G)

Even without a flow graph, V(G) can be determined by counting the number of conditional statements in code and adding 1 to it.

Prepare test cases. Each test case is executed and compared to the expected results.

Graph Matrices

A graph matrix is a square matrix whose rows and columns are equal to the number of nodes in the flow graph. Each row and column identifies a particular node and matrix entries represent a connection between the nodes.

The following points describe a graph matrix:

Each cell in the matrix can be a direct connection or link between one node to another node.

If there is a connection from node 'a' to node 'b', then it does not mean that there is connection from node 'b' to node 'a'.

Conventionally, to represent a graph matrix, digits are used for nodes and letter symbols for edges or connections.

Connection matrices

The links between two nodes are assigned a link weight, which becomes the entry in the cell of the matrix.

when the connection exists, then the link weight is 1.

A matrix defined with link weights is called a connection matrix.

Black Box Testing

Test the functionality of an application without knowing its internal structure, coding information and knowledge of internal paths of the software.

Test cases are built based on what the application is supposed to do.

Also known as behavioral testing or specification testing.

Focused on input to the software and output from the software.

Black box testing can be applied during each level of the software testing process.

Testers select valid test data for positive test cases.

Also the tester selects invalid test data for negative test scenarios.

Defect is detected if the actual result is not the same as expected result.

Done using different testing techniques like Boundary Value Analysis (BVA), equivalence class decision models etc.

Methods for developing test cases

i. Boundary Value Analysis (BVA)

It is a process of testing boundaries of the input values.

It is the most commonly used technique in Black Box Testing.

Basic idea is to select input values : minimum, just above minimum, normal value, maximum value and just below maximum value.

It checks for the input values near the boundary that have a higher chance of error.

ii. Equivalence class partitioning

Reduces the number of all possible inputs by dividing them into classes.

Test the application thoroughly and avoid redundancy of input values.

BVA and equivalence partitioning are closely related and used together.

iii. Graph Based

This technique of Black box testing involves a graph drawing that depicts the link between the causes (inputs) and the effects (output), which trigger the effects.

Cause-Effect graph technique is based on a collection of requirements and used to determine minimum possible test cases which can cover a maximum test area of the software.

The main advantage of cause-effect graph testing is, it reduces the time of test execution and cost.

Graph Representation

A collection of nodes that represent objects,

Links that represent the relationships between objects,

Node weights that describe the properties of a node (e.g., a specific data value or state behavior),

Link weights that describe some characteristics of a link.

Nodes are represented as circles connected by links that take a number of different forms.

A directed link (represented by an arrow) indicates that a relationship moves in only one direction.

A bidirectional link, also called a symmetric link, implies that the relationship applies in both directions.

Parallel links are used when a number of different relationships are established between graph nodes.

iv. Error- Guessing

This method makes use of the tester's experience and skill for testing similar applications to find defects which may not be determined by formal techniques.

It is solely based on judgment and perception of the earlier end user experience.

It is usually done after formal testing techniques are applied.

Types of Black Box Testing

- Functional testing

- Non-Functional Testing
- Regression Testing

Advantages

Efficient for large systems.

Identifies contradictions in functional specifications.

Detailed functional knowledge of the system is not a prerequisite for testers.

Testers and developers work immediately.

Disadvantages

Difficult to find out all possible inputs for test cases in limited time

Test method cannot be used for complex code

Test cases cannot be designed without knowledge of functional spec.