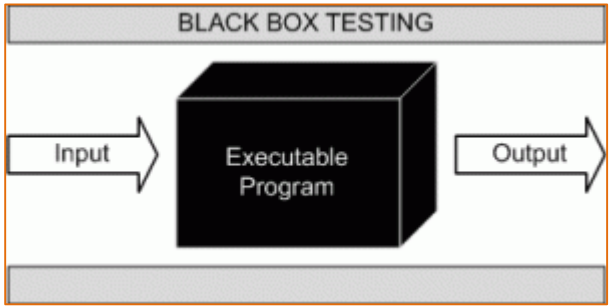
Unit- III
Black Box Testing

- Dr. Shivani Budhkar

Black Box Testing

- It is defined as a testing method in which functionality of the Application Under Test (AUT) is tested without looking at the internal code structure, implementation details and knowledge of internal paths of the software.
- This type of testing is based entirely on software requirements and specifications.
- In Black Box Testing we just focus on inputs and output of the software system without bothering about internal knowledge of the software program.

OR



Why Black Box Testing

- Black box testing is done based on requirements
- Black box testing addresses the stated requirements as well as implied requirements
- Black box testing encompasses the end user perspective
- Black box testing handles valid and invalid inputs

How to do Black box testing

The various techniques:

- Requirements based Testing
- Positive and Negative testing
- Boundary Value analysis
- Equivalence Partitioning
- State based testing
- Compatibility testing
- User Documentation Testing
- Domain Testing

Requirements based Testing

- Deals with validating requirements
- Not all requirements are explicitly stated, some of requirements are implicit
- Detailed review of requirements is done
- Requirements are tracked by RTM
- The main purpose of RTM to see that all test cases are covered so that no functionality should be miss while testing
- RTM helps in identifying relationship between the requirements and test cases.
- Thus an RTM plays a valuable role in requirements based engineering

Sample Requirements Traceability matrix

Req.ID	Description (Functional specification)	Priority (H,M,L)	Test conditions(Test scenario ID)	Test case IDs	Phase of testing	Status	Defect Id
BR-01	Loan Process-New User	H	TS-Loan-001 “Apply Loan” feature	TC_New User-01 TC_New User-02	System testing	Pass	None

Sample Test execution data

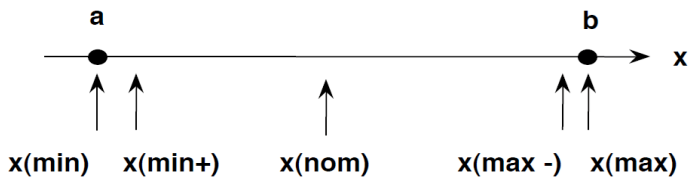
Sr. No	Req.ID	Description (Functional specification)	Priority (H,M,L)	Test case IDs	Total test cases	Test cases passed	Test cases failed	% Pass	No. of defects
1	BR-01	Loan Process- New User	H	TC_New User-01 TC_New User-02	2	2	0	100	0
…	…	…	…	…	…	…	…	…	…
….	…	…	…	…	…	…	…	…	…
Total	8				12	10	2	83	12

Positive and Negative Testing

- Tries to prove that given product does what it is supposed to do
- When test case verifies the requirements of the product with a set of expected output, it is called positive test case.
- The purpose of positive testing is to prove that the product works as per specification and expectations
- Negative testing is done to show that the product does not fail when an unexpected input is given
- The propose of negative testing is to try and break the system.
- Negative testing covers scenarios for which the product is not designed and coded
- Negative testing is generally to ensure the stability of application

Boundary value analysis(BVA)

- Boundary testing is the process of testing between extreme ends or boundaries between partitions of the input values.
- So these extreme ends like Start- End, Lower- Upper, Maximum-Minimum, Just Inside-Just Outside values are called boundary values and the testing is called "boundary testing".
- The basic idea in boundary value testing is to select input variable values at their:
  - Minimum
  - Just above the minimum
  - A nominal value
  - Just below the maximum
  - Maximum



- BVA is useful to generate test cases when the input (or output)data is made up of clearly identifiable boundaries or ranges

Example BVA

- **Test cases for input box accepting numbers between 1 and 1000 using Boundary value analysis:**
- **#1)** Test cases with test data exactly as the input boundaries of input domain i.e. values 1 and 1000 in our case.
- **#2)** Test data with values just below the extreme edges of input domains i.e. values 0 and 999.
- **#3)** Test data with values just above the extreme edges of the input domain i.e. values 2 and 1001.

Equivalence Class Partitioning (ECP)

- Equivalent Class Partitioning is a black box technique which can be applied to all levels of testing like unit, integration, system, etc.
- In this technique, you divide the set of test condition into a partition that can be considered the same.
- It divides the input data of software into different equivalence data classes.
- You can apply this technique, where there is a range in the input field.
- This technique involves-
  - Identifying all partitions for the complete set of inputs, output values for a product
  - Picking up one member value from each partition for testing to maximize complete coverage

Example ECP

- If you are testing for an input box accepting numbers from 1 to 1000 then there is no use in writing thousand test cases for all 1000 valid input numbers plus other test cases for invalid data.
- Using the Equivalence Partitioning method test cases can be divided into three sets of input data called classes. Each test case is representative of a respective class.
- So in the example, we can divide our test cases into three equivalence classes of some valid and invalid inputs.

Example ECP

- **Test cases for input box accepting numbers between 1 and 1000 using Equivalence Partitioning:**
- **#1)** One input data class with all valid inputs. Pick a single value from range 1 to 1000 as a valid test case. If you select other values between 1 and 1000 the result is going to be the same. So one test case for valid input data should be sufficient.
- **#2)** Input data class with all values below the lower limit. i.e. any value below 1, as an invalid input data test case.
- **#3)** Input data with any value greater than 1000 to represent the third invalid input class.
- So using Equivalence Partitioning you have categorized all possible test cases into three classes. Test cases with other values from any class should give you the same result.
- We have selected one representative from every input class to design our test cases. Test case values are selected in such a way that largest number of attributes of equivalence class can be exercised.
- Equivalence Partitioning uses fewest test cases to cover maximum requirements.

Test Cases using ECP & BVA

Test case ID	Partition	Type of input	Test Data	Expected Output
TC_001	Number between 1 to 1000	Valid	1,2, 1000, 110,999	Number Accepted
TC_002	Number below 1	Invalid	0	Error Message
TC_003	Number greater than 1000	Invalid	1001,1003	Error Message

State Based or Graph based testing

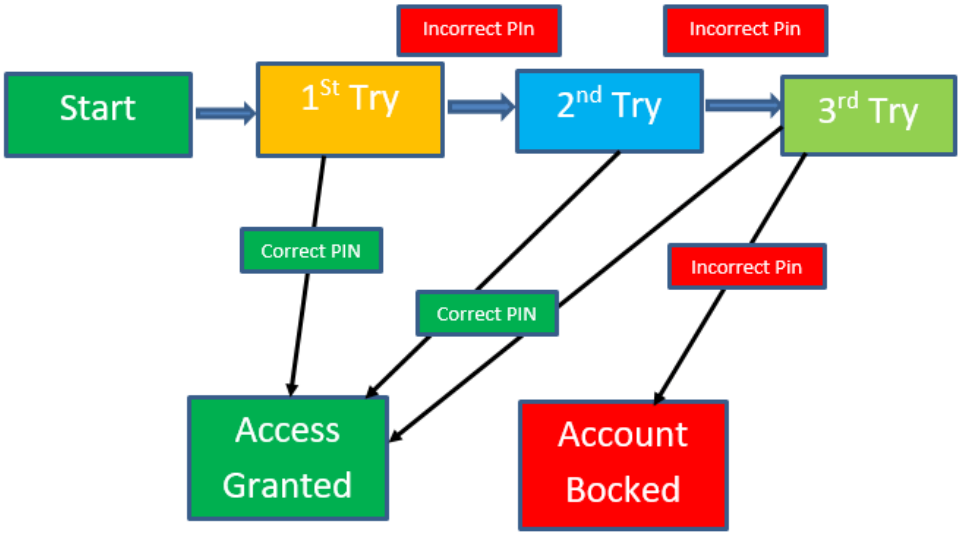
- It is a black box testing technique in which the tester analyses the behaviour of an application under test for different input conditions in a sequence.
- In this technique, tester provides both positive and negative input test values and record the system behaviour.
- Useful to generate test cases for state machine as it has dynamic behavior
- It is helpful where you need to **test different system transitions.**
- Graph based testing methods are applicable to generate test cases for state machines such as language translators ,workflows, Transaction flows, and data flows

State Based or Graph based testing

- When to Use State Transition?
- This can be used when a tester is testing the application for a finite set of input values.
- When the tester is trying to test sequence of events that occur in the application under test. I.e., this will allow the tester to test the application behaviour for a sequence of input values.
- When the system under test has a dependency on the events/values in the past.
- Four parts of state transition diagram
  - States
  - Transitions
  - Events
  - Action

State Based or Graph based testing- Example

- Let's consider an ATM system function where if the user enters the invalid password three times the account will be locked
- In this system, if the user enters a valid password in any of the first three attempts the user will be logged in successfully. If the user enters the invalid password in the first or second try, the user will be asked to re-enter the password. And finally, if the user enters incorrect password 3rd time, the account will be blocked.



State Based or Graph based testing- Example

- In the diagram whenever the user enters the correct PIN he is moved to Access granted state, and if he enters the wrong password he is moved to next try and if he does the same for the 3rd time the account blocked state is reached.
- **State Transition Table**

	Correct PIN	Incorrect PIN
S1) Start	S5	S2
S2) 1st attempt	S5	S3
S3) 2nd attempt	S5	S4
S4) 3rd attempt	S5	S6
S5) Access Granted	-	-
S6) Account blocked	-	-

- In the table when the user enters the correct PIN, state is transitioned to S5 which is Access granted. And if the user enters a wrong password he is moved to next state. If he does the same 3rd time, he will reach the account blocked state.

State Based or Graph based testing

- Advantages and Disadvantages of State Transition Technique

Advantages	Disadvantages
This testing technique will provide a pictorial or tabular representation of system behaviour which will make the tester to cover and understand the system behaviour effectively.	The main disadvantage of this testing technique is that we can't rely in this technique every time. For example, if the system is not a finite system (not in sequential order), this technique cannot be used.
By using this testing, technique tester can verify that all the conditions are covered, and the results are captured	Another disadvantage is that you have to define all the possible states of a system. While this is all right for small systems, it soon breaks down into larger systems as there is an exponential progression in the number of states.

Compatibility Testing

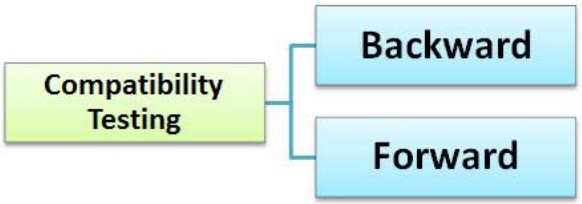
- Testing is done to ensure that the product features work consistently with different infrastructures components is called compatibility testing
- When infrastructure parameters are changes, the product is expected to still behave correctly and produce the desired or expected results
- The infrastructure parameters could be of hardware, software or other components
- Definition - Compatibility Testing is a type of Software testing to check whether your software is capable of running on different hardware, operating systems, applications, network environments or Mobile devices.
- Compatibility Testing is a type of Non-functional testing
- **Types of Compatibility Tests**
  - Hardware
  - OS
  - Software
  - Network
  - Browser
  - Devices
  - Mobile
  - Versions

Compatibility Testing

- **Hardware**: It checks software to be compatible with different hardware configurations.
- **Operating Systems**: It checks your software to be compatible with different Operating Systems like Windows, Unix, Mac OS etc.
- **Software**: It checks your developed software to be compatible with other software. For example, MS Word application should be compatible with other software like MS Outlook, MS Excel, VBA etc.
- **Network:** Evaluation of performance of a system in a network with varying parameters such as Bandwidth, Operating speed, Capacity. It also checks application in different networks with all parameters mentioned earlier.
- **Browser**: It checks the compatibility of your website with different browsers like Firefox, Google Chrome, Internet Explorer etc.
- **Devices**: It checks compatibility of your software with different devices like USB port Devices, Printers and Scanners, Other media devices and Blue tooth.
- **Mobile**: Checking your software is compatible with mobile platforms like Android, iOS etc.
- **Versions of the software:** It is verifying your software application to be compatible with different versions of the software. For instance checking your Microsoft Word to be compatible with Windows 7, Windows 7 SP1, Windows 7 SP2, Windows 7 SP3

Compatibility Testing

- Two types of compatibility checking



- **Backward compatibility Testing** is to verify the behaviour of the developed hardware/software with the **older versions** of the hardware/software.
- **Forward compatibility Testing** is to verify the behaviour of the developed hardware/software with the **newer versions** of the hardware/software.

User Documentation Testing

- User documentation is done to ensure the documentation matches the product and vice versa
- Test documentation is documentation of artifacts created before or during the testing of software.
- It helps the testing team to estimate testing effort needed, test coverage, resource tracking, execution progress, etc.
- It is a complete suite of documents that allows you to describe and document test planning, test design, test execution, test results that are drawn from the testing activity.
- **Objectives**

  1. To check if what is stated in the document is available in the product

  2. To check if what is there in the product is explained correctly in the document
- Testing activities generally consume 30% to 50% of software development project effort. Documentations help to identify Test process improvement that can be applied to future projects.

Examples of Test Documentation

Types of Testing	Description
Test policy	It is a high-level document which describes principles, methods and all the important testing goals of the organization.
Test strategy	A high-level document which identifies the Test Levels (types) to be executed for the project.
Test plan	A test plan is a complete planning document which contains the scope, approach, resources, schedule, etc. of testing activities.
Requirements Traceability Matrix	This is a document which connects the requirements to the test cases.
Test Scenario	Test scenario is an item or event of a software system which could be verified by one or more Test cases.
Test case	It is a group of input values, execution preconditions, expected execution postconditions and results. It is developed for a Test Scenario.
Test Data	Test Data is a data which exists before a test is executed. It used to execute the test case.
Defect Report	Defect report is a documented report of any flaw in a Software System which fails to perform its expected function.
Test summary report	Test summary report is a high-level document which summarizes testing activities conducted as well as the test result.

Advantages of User Documentation testing

- The main reason behind creating test documentation is to either reduce or remove any uncertainties about the testing activities. Helps you to remove ambiguity which often arises when it comes to the allocation of tasks
- Documentation not only offers a systematic approach to software testing, but it also acts as training material to freshers in the software testing process
- It is also a good marketing & sales strategy to showcase Test Documentation to exhibit a mature testing process
- Test documentation helps you to offer a quality product to the client within specific time limits
- In Software Engineering, Test Documentation also helps to configure or set-up the program through the configuration document and operator manuals
- Test documentation helps you to improve transparency with the client

Disadvantages of User Documentation testing

- The cost of the documentation may surpass its value as it is very time-consuming
- Many times, it is written by people who can't write well or who don't know the material
- Keeping track of changes requested by the client and updating corresponding documents is tiring.
- Poor documentation directly reflects the quality of the product as a misunderstanding between the client and the organization can occur

Domain Testing

- It can be considered as next level of testing in which we do not look even at the specifications of the software product but are testing the product purely based on domain knowledge and expertise in the domain of application under test.
- Requires business domain knowledge rather than knowledge of specification
- Can be considered as extension of black box testing
- It exploits the tester's domain knowledge to test the suitability of the product to what the users do on a typical day.

Context of white box, black box and domain testing