**Module-2**

* **What is Exploratory Testing?**

Exploratory testing is a concurrent process where Test design, execution and logging happen simultaneously Testing is often not recorded Makes use of experience, heuristics and test patterns Testing is based on a test charter that may include Scope of the testing (in and out) The focus of exploratory testing is more on testing as a “thinking” activity. A brief description of how tests will be performed Expected problems Is carried out in time boxed intervals.

**• What is traceability matrix?**

A Traceability Matrix is a document that co-relates any two-baseline documents that require a many-to-many relationship to check the completeness of the relationship.

It is used to track the requirements and to check the current project requirements are met.

**• What is Boundary value testing?**

Boundary value analysis is a methodology for designing test cases that concentrates software testing effort on cases near the limits of valid ranges.

Boundary value analysis is a method which refines equivalence partitioning.

Boundary value analysis generates test cases that highlight errors better than equivalence partitioning.

The trick is to concentrate software testing efforts at the extreme ends of the equivalence classes.

At those points when input values change from valid to invalid errors are most likely to occur.

Boundary Value Analysis (BVA) uses the same analysis of partitions as EP and is usually used in conjunction with EP in test case design

**• What is Equivalence partitioning testing?**

Aim is to treat groups of inputs as equivalent and to select one representative input to test them all EP can be used for all Levels of Testing

Equivalence partitioning is the process of defining the optimum number of tests by:

Reviewing documents such as the Functional Design Specification and Detailed Design Specification, and identifying each input condition within a function, Selecting input data that is representative of all other data that would likely invoke the same process for that particular condition.

The numbers fall into a partition where each would have the same, or equivalent, result i.e. an Equivalence Partition (EP) or Equivalence Class

EP says that by testing just one value we have tested the partition (typically a mid-point value is used). It assumes that:

If one value finds a bug, the others probably will too

If one doesn't find a bug, the others probably won't either

In EP we must identify Valid Equivalence partitions and Invalid Equivalence partitions where applicable (typically in range tests)

**• What is Integration testing?**

Integration Testing - Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems.

Integration Testing is a level of the software testing process where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing.

It tests integration or interfaces between components, interactions to different parts of the system such as an operating system, file system and hardware or interfaces between systems.

It is done by a specific integration tester or test team.

Components may be code modules, operating systems, hardware and even complete systems.

There are 2 levels of Integration Testing

* Component Integration Testing
* System Integration Testing

**• What determines the level of risk?**

The likelihood of an adverse event and the impact of the event.

**• What is Alpha testing?**

Alpha testing is the initial phase of validating whether a new product will perform as expected. Alpha tests are carried out early in the development process by internal staff and are followed up with Beta testing, in which a sampling of the intended audience actually tries the product out.

**• What is beta testing?**

Beta Testing is one of the Acceptance Testing types, which adds value to the product as the end-user (intended real user) validates the product for functionality, usability, reliability, and compatibility.

Inputs provided by the end-users help in enhancing the quality of the product further and lead to its success. This also helps in decision making to invest further in future products or the same product for improvisation.

**• What is component testing?**

Component(Unit) – A minimal software item that can be tested in isolation. It means “A unit is the smallest testable part of software.” Component Testing – The testing of individual software components.

Unit Testing is a level of the software testing process where individual units/components of a software/system are tested. The purpose is to validate that each unit of the software performs as designed. Tests are typically written and run by software developers to ensure that code meets its design and behaves as intended with debugging tool. It is performed by using the White Box Testing method.

**• What is functional system testing?**

Functional System Testing : A requirement that specifies a function that a system or system component must perform.

A Requirement may exist as a text document and/or a model.

There is two types of Test Approach

* Requirement Based Functional Testing
* Process Based Testing

Functional System Testing Functionality As below:

* Accuracy: Provision of right or agreed results or effects
* Interoperability: Ability to interact with specified systems
* Compliance: Adhere to applicable standards, conventions, regulations
* or laws
* Auditability: Ability to provide adequate and accurate audit data
  + - Suitability Presence and appropriateness of functions for specified
    - tasks

**• What is Non-Functional Testing?**

Non-Functional Testing: Testing the attributes of a component or system that do not relate to functionality, e.g. reliability, efficiency, usability, interoperability, maintainability and portability.

- May be performed at all Test levels (not just Non Functional System Testing)

- Measuring the characteristics of the system/software that can be

quantified on a varying scale- e.g. performance test scaling

- Non-functional testing includes, but is not limited to, performance

testing, load testing, stress testing, usability testing, maintainability

testing, reliability testing and portability testing.

**• What is GUI Testing?**

GUI of the System under Test. GUI testing involves checking the screens

with the controls like menus, buttons, icons, and all types of bars – tool bar,

menu bar, dialog boxes and windows etc.

WHAT DO YOU CHECK IN GUI TESTING?

* Check all the GUI elements for size, position, width, length and acceptance of characters or numbers. For instance, you must be able to provide inputs to the input fields.
* Check you can execute the intended functionality of the application using the GUI
* Check Error Messages are displayed correctly
* Check for Clear demarcation of different sections on screen
* Check Font used in application is readable
* Check the alignment of the text is proper
* Check the Color of the font and warning messages is aesthetically pleasing
* Check that the images have good clarity
* Check that the images are properly aligned
* Check the positioning of GUI elements for different screen resolution.

Approach of GUI Testing

MANUAL BASED TESTING

* Under this approach, graphical screens are checked manually by testers
* in conformance with the requirements stated in business requirements
* document.

RECORD AND REPLAY

* GUI testing can be done using automation tools. This is done in 2 parts. During
* Record , test steps are captured into the automation tool. During playback, the
* recorded test steps are executed on the Application under Test. Example of
* such tools - QTP.

MODEL BASED TESTING

* A model is a graphical description of system’s behavior. It helps us to
* understand and predict the system behavior. Models help in a generation of
* efficient test cases using the system requirements

GUI Testing Examples

**Web Based Testing & Desktop Based Testing :**

* The scrollbar should be enabled only when necessary.
* Font size, style, and color for headline, description text, labels, infield data, and grid info should

be standard as specified in SRS.

* The description text box should be multi-lined.
* Enough space should be provided between field labels, columns, rows, error messages, etc.

**Mobile Based Testing :**

* If mobile is in every orientation mode so display image , video properly.
* Every app will display in responsive type .
* Alignment should be apply properly of every field.

**Game Based Testing :**

* Game infra design will showing properly
* Game points or score will display proper with its background color.
* Game sound manage with its background effect
* Can be also conducted in advance of designing page layouts or navigation menus

**• What is Adhoc testing?**

Adhoc testing is an informal testing type with an aim to break the system.It does not follow any test design techniques to create test cases.

This testing is primarily performed if the knowledge of testers in the system under test is very high. Testers randomly test the application without any test cases or any business requirement document.

Adhoc Testing does not follow any structured way of testing and it is randomly done on any part of application. Main aim of this testing is to find defects by random checking. Adhoc testing can be achieved with the testing technique called Error Guessing.

Error guessing can be done by the people having enough experience on the system to “guess” the most likely source of errors. The Error guessing is a technique where the experienced and good testers are encouraged to think of situations in which the software may not be able to cope. Some people seem to be naturally good at testing and others are good testers because they have a lot of experience either as tester or working with a particular system and so are able to find out its weaknesses. This is why an error guessing approach, used after more formal techniques have been applied to some extent, can be very effective. It also saves a lot of time because of the assumptions and guessing made by the experienced testers to find out the defects which otherwise won’t be able to find.

There are different types of Adhoc testing and they are listed as below:

1. Buddy Testing

- Two buddies mutually work on identifying defects in the same module.

Mostly one buddy will be from development team and another person will be from testing team. Buddy testing helps the testers develop better test cases and development team can also make design changes early. This testing usually happens after unit testing completion.

2. Pair testing

- Two testers are assigned modules, share ideas and work on the same machines to find defects. One person can execute the tests and another person can take notes on the findings. Roles of the persons can be a tester and scriber during testing.

3. Monkey Testing

- Randomly test the product or application without test cases with a goal to break the system.

**• What is load testing?**

Its a performance testing to check system behavior under load.

Testing an application under heavy loads, such as testing of a web site under a range of loads to determine at what point the system’s response time degrades or fails.

Load testing is a kind of performance testing which determines a system’s performance under real-life load conditions. This testing helps determine how the application behaves when multiple users access it simultaneously.

This testing usually identifies –

* The maximum operating capacity of an application
* Determine whether current infrastructure is sufficient to run the application
* Sustainability of application with respect to peak user load
* Number of concurrent users that an application can support, and scalability to allow more users to access it.
* It is a type of non-functional testing. Load testing is commonly used for the Client/Server, Web based applications – both Intranet and Internet.

**• What is stress Testing?**

System is stressed beyond its specifications to check how and when it fails. Performed under heavy load like putting large number beyond storage capacity, complex database queries, continuous input to system or database load.

Stress testing is used to test the stability & reliability of the system. This test mainly determines the system on its robustness and error handling under extremely heavy load conditions.

- It even tests beyond the normal operating point and evaluates how the system works under those extreme conditions.

- Stress Testing is done to make sure that the system would not crash under crunch situations.

- Stress testing is also known as endurance testing.

**• What is white box testing and list the types of white box testing?**

White Box Testing: Testing based on an analysis of the internal structure of the component or system.

- Structure-based testing technique is also known as ‘white-box’ or ‘glass-box’ testing technique because here the testers require knowledge of how the software is implemented, how it works.

- In white-box testing the tester is concentrating on how the software does it.

- For example, a structural technique may be concerned with exercising loops in the software.

- Different test cases may be derived to exercise the loop once, twice, and many times. This may be done regardless of the functionality of the software.

- Structure-based techniques are also used in system and acceptance testing, but the structures are different.

- For example, the coverage of menu options or major business transaction could be the structural element in system or acceptance testing.

Types of the white box testing

* Web Based Testing
* Desktop Based Testing
* Game Based Testing
* Branch Condition testing
* Branch Condition Combination testing
* Modified Condition Decision testing
* Dataflow testing
* Linear Code Sequence And Jump (LCSAJ) testing

**• What is black box testing? What are the different black box testing techniques?**

It is a testing, either functional or non-functional, without reference to the internal structure of the component or system.

Specification-based testing technique is also known as ‘black-box’ or input/output driven testing techniques because they view the software as a black-box with inputs and outputs.

The testers have no knowledge of how the system or component is structured inside the box. In black-box testing the tester is concentrating on what the software does, not how it does it.

Specification-based techniques are appropriate at all levels of testing (component testing through to acceptance testing) where a specification exists.

- For example, when performing system or acceptance testing, the requirements specification or functional specification may form the basis of the tests.

There are four specification-based or black-box

technique:

* Equivalence partitioning
* Boundary value analysis
* Decision tables
* State transition testing
* Use-case Testing
* Other Black Box Testing

- Syntax or Pattern Testing

**• Mention what are the categories of defects?**

Data Quality/Database Defects: Deals with improper handling of data in the database.

Examples:

* Values not deleted/inserted into the database properly
* Improper/wrong/null values inserted in place of the actual values

Critical Functionality Defects: The occurrence of these bugs hampers the crucial functionality of the application.

Examples: - Exceptions

Functionality Defects: These defects affect the functionality of the application.

Examples:

* All JavaScript errors
* Buttons like Save, Delete, Cancel not performing their intended functions
* A missing functionality (or) a feature not functioning the way it is intended to
* Continuous execution of loops

Security Defects: Application security defects generally involve improper handling of data sent from the user to the application. These defects are the most severe and given highest priority for a fix.

Examples:

* Authentication: Accepting an invalid username/password
* Authorization: Accessibility to pages though permission not given

User Interface Defects: As the name suggests, the bugs deal with problems related to UI are usually considered less severe.

Examples:

* Improper error/warning/UI messages
* Spelling mistakes
* Alignment problems

**• Mention what big bang testing is?**

In Big Bang integration testing all components or modules is integrated simultaneously, after which everything is tested as a whole.

Big Bang testing has the advantage that everything is finished before integration testing starts.

The major disadvantage is that in general it is time consuming and difficult to trace the cause of failures because of this late integration. Here all component are integrated together at **once**, and then

tested.

**• What is the purpose of exit criteria?**

Purpose of exit criteria is to define when we STOP testing either at the:

* End of all testing – i.e. product Go Live
* End of phase of testing (e.g. hand over from System Test to UAT)

**• When should "Regression Testing" be performed?**

* Change in requirements and code is modified according to the requirement
* New feature is added to the software
* Defect fixing
* Performance issue fix

**• What is 7 key principles? Explain in detail?**

1. Testing shows presence of defects
2. Exhaustive testing is not possible
3. Early testing
4. Defect clustering
5. Pesticide paradox
6. Testing is context dependent
7. Absence of errors fallac

1. Testing shows presence of Defects

Testing reduces the probability of undiscovered defects remaining in the software but, even if no defects are found, it is not a proof of correctness. We test to find Faults. As we find more defects, the probability of undiscovered defects remaining in a system reduces. However Testing cannot prove that there are no defects present.

2. Exhaustive Testing is Impossible!

Testing everything including all combinations of inputs and preconditions is not possible.

So, instead of doing the exhaustive testing we can use risks and

priorities to focus testing efforts.

For example: In an application in one screen there are 15 input fields, each having 5 possible values, then to test all the valid combinations you would need 30 517 578 125 (515) tests. This is very unlikely that the project timescales would allow for this number of tests.

So, accessing and managing risk is one of the most important activities and reason for testing in any project.

We have learned that we cannot test everything (i.e. all combinations of inputs and pre-conditions).

That is we must Prioritise our testing effort using a Risk Based Approach.

3. Early Testing

Testing activities should start as early as possible in the software or system development life cycle, and should be focused on defined objectives.

Testing activities should start as earlyas possible in the development life cycle

These activities should be focused on defined objectives – outlined in the Test Strategy.

4. Defect Clustering

* A small number of modules contain most of the defects discovered during pre-release testing, or are responsible for the most operational failures.
* Defects are not evenly spread in a system
* They are ‘clustered’
* In other words, most defects found during testing are usually confined to a small number of modules
* Similarly, most operational failures of a system are usually confined to a small number of modules

5. Pesticide Paradox

* If the same tests are repeated over and over again, eventually the same set of test cases will no longer find any new defects.
* To overcome this “pesticide paradox”, the test cases need to be regularly reviewed and revised, and new and different tests need to be written to exercise different parts of the software or system to potentially find more defects.
* Testing identifies bugs, and programmers respond to fix them.
* As bugs are eliminated by the programmers, the software improves.
* As software improves the effectiveness of previous tests erodes.

6. Testing is Context Dependent

* Testing is basically context dependent.
* Testing is done differently in different contexts
* Different kinds of sites are tested differently.
* For example
* Safety – critical software is tested differently from an e-commerce site.
* Whilst, Testing can be 50% of development costs, in NASA's Apollo program it was 80% testing
* 3 to 10 failures per thousand lines of code (KLOC) typical for commercial software
* 1 to 3 failures per KLOC typical for industrial software
  + failures per KLOC for NASA Shuttle code!
* Also different industries impose different testing standards

7. Absence of Errors Fallacy

* If the system built is unusable and does not fulfill the user’s needs and expectations then finding and fixing defects does not help.
* If we build a system and, in doing so, find and fix defects ....
* It doesn’t make it a good system
* Even after defects have been resolved it may still be unusable and/or does not fulfil the users’ needs and expectations.

**• Difference between QA v/s QC v/s Tester**

|  |  |  |
| --- | --- | --- |
| QA | QC | Tester |
| Activities which ensure the  implementation of processes,  procedures and standards in  context to verification of  developed software and intended  requirements. | Activities which ensure the  verification of developed  software with respect to  documented (or not in some  cases) requirements. | Activities which ensure  the identification of  bugs/error/defects in the  Software. |
| Focuses on processes and  procedures rather than conducting  actual testing on the system. | Focuses on actual testing by  executing Software with intend  to identify bug/defect through  implementation of procedures  and process. | Focuses on actual testing. |
| Process oriented activities. | Product oriented activities. | Product oriented activities. |
| Preventive activities. | It is a corrective process. | It is a preventive process. |
| It is a subset of Software Test Life  Cycle (STLC). | QC can be considered as the  subset of Quality Assurance. | Testing is the subset of  Quality Control. |

**• Difference between Smoke and Sanity?**

|  |  |
| --- | --- |
| Smoke Testing is performed to ascertain that the critical functionalities of the program is working fine | Sanity Testing is done to check the new functionality and bugs have been fixed. |
| This testing is perform by developers or testers | Sanity testing is usually performed by testers |
| Smoke testing is usually documented or scripted | Sanity testing is usually not documented and unscripted |
| Smoke testing is a subset of Regression testing | Sanity testing is a subset of Acceptance testing |
| Smoke testing exercises the entire system from end to end | Sanity testing exercises only the particular component of the entire system |

**• Difference between verification and ValidationCribjectist**

|  |  |  |
| --- | --- | --- |
| Criteria | Verification | Validation |
| Definition | The process of evaluating work-products (not the actual  final product) of a development phase to determine whether  they meet the specified requirements for that phase. | The process of evaluating software during or at the end of the development process to determine whether it satisfies specified business requirements. |
| Objective | To ensure that the product is being built according to the requirements and design specifications. In other words, to ensure that work products meet their specified requirements. | To ensure that the product actually meets the user’s needs, and that the  specifications were correct in the first place. In other words, to demonstrate that the product fulfills its intended use when placed in its intended environment. |
| Question | Are we building the product right? | Are we building the right product? |
| Evaluation items | Plans, Requirement Specs, Design  Specs, Code, Test Cases | The actual product/software |
| Activities | Reviews  ∙ Walkthroughs  ∙ Inspections | Testing |

**Evaluation**

**• Explain types of Performance testing.**

* Load testing
* Stress testing
* Endurance testing
* Spike testing
* Volume testing
* Scalability testing

**• What is Error, Defect, Bug and failure?**

A mistake in coding is called error, error found by tester is called defect, defect accepted by development team then it is called bug, build does not meet the requirements then it is

failure.

**• Difference between Priority and Severity**

* **Priority** is Relative and Business-Focused. It defines the order in which we should resolve a defect. Should we fix it now, or can it wait? This priority status is set by the tester to the developer mentioning the time frame to fix the defect.
* If high priority is mentioned then the developer has to fix it at
* the earliest.
* The priority status is set based on the customer requirements.

For example: If the company name is misspelled in the home page of the website, then the priority is high and severity is low to fix it.

Priority can be of following types:

Low: The defect is an irritant which should be repaired, but repair can be deferred until after more serious defect has been fixed.

Medium: The defect should be resolved in the normal course of development activities. It can wait until a new build or version is created.

High: The defect must be resolved as soon as possible because the defect is affecting the application or the product severely. The system cannot be used until the repair has been done.

Critical: Extremely urgent, resolve immediately

* **Severity** is absolute and Customer-Focused.
* It is the extent to which the defect can affect the software.
* In other words it defines the impact that a given defect has on the system.

For example: If an application or web page crashes when a remote link is clicked, in this case clicking the remote link by an user is rare but the impact of application crashing is severe. So the severity is high but

priority is low.

Severity can be of following types:

Critical: The defect that results in the termination of the complete system or one or more component of the system and causes extensive corruption of the data. The failed function is unusable and there is no acceptable alternative method to achieve the required results then the severity will be

stated as critical.

Major (High): The defect that results in the termination of the complete system or one or more component of the system and causes extensive corruption of the data. The failed function is unusable but there exists an acceptable alternative method to achieve the required results then the severity will be stated as major.

Moderate (Medium): The defect that does not result in the termination, but causes the system to produce incorrect, incomplete or inconsistent results then the severity will be stated as moderate.

Minor (Low): The defect that does not result in the termination and does not damage the usability of the system and the desired results can be easily obtained by working around the defects then the severity is stated as minor.

Cosmetic: The defect that is related to the enhancement of the system where the changes are related to the look and field of the application then the severity is stated as cosmetic.

**• What is Bug Life Cycle?**

The duration or time span between the first time defects is found and the time that it is closed successfully, rejected, postponed or deferred is called as ‘Defect Life Cycle’.

**• Explain the difference between Functional testing and Non Functional testing**

|  |  |
| --- | --- |
| **Functional testing** | **Non functional testing** |
| Performed using the functional specification provided by client and verifies the system against the functional requirement. | Nonfunctional Testing checks the Performance,  reliability, scalability and other non-functional aspects. |
| Functional testing is executed first | Non functional testing should be performed  after functional testing |
| Manual testing or automation tools can be  used for functional testing | Using tools will be effective for this testing |
| Business requirements are the inputs to functional testing | Performance parameters like speed , scalability are  inputs to non-functional testing. |
| Functional testing describes what the product does | Nonfunctional testing describes how good the  product works |
| Easy to do manual testing | Tough to do manual testing |
| Types of Functional testing are  ∙ Unit Testing  ∙ Smoke Testing  ∙ Sanity Testing  ∙ Integration Testing  ∙ White box testing  ∙ Black Box testing  ∙ User Acceptance testing  ∙ Regression Testing | Types of Nonfunctional testing are  ∙ Performance Testing  ∙ Load Testing  ∙ Volume Testing  ∙ Stress Testing  ∙ Security Testing  ∙ Installation Testing  ∙ Penetration Testing  ∙ Compatibility Testing  ∙ Migration Testing |

**• What is the difference between the STLC (Software Testing Life Cycle) and SDLC (Software Development Life Cycle)?**

STLC and SDLC are both interrelated in some way, or you could say that one is the predecessor to the other. SDLC refers to a sequence of activities during the software development process, whereas STLC refers to a sequence of activities during software testing. The SDLC is primarily concerned with software development.

**• What is the difference between test scenarios, test cases, and test script?**

**A test script** in software testing is a set of instructions that will be

performed on the system under test to test that the system functions as

expected.

There are various means for executing test scripts.

* Manual Testing
* Automation Testing

**A Scenario** is any functionality that can be tested. It is also called Test Condition, or Test Possibility.

* Test Scenario is ‘What to be tested’
* Test scenario is nothing but test procedure.
* The scenarios are derived from use cases.
* Test Scenario represents a series of actions that are associated together. Scenario is thread of operations

**Test cases** involve the set of steps, conditions and inputs which can be used while performing the testing tasks.

* Test Case is ‘How to be tested’
* Test case consist of set of input values, execution precondition, expected Results and executed post-condition developed to cover certain test Condition.
* Test cases are derived (or written) from test scenario.
* Test Case represents a single (low level) action by the user.
* Test cases are set of input and output given to the System.

• **Explain what Test Plan is? What is the information that should be covered.**

A Test Plan is a detailed document that catalogs the test strategies, objectives, schedule, estimations, deadlines, and resources required to complete that project.

**• What are the different Methodologies in Agile Development Model?**

 Scrum, Kanban, Extreme Programming (XP), Lean Development e Crystal.

**• Explain the difference between Authorization and Authentication in Web testing. What are the common problems faced in Web testing?**

**Authentication verifies the identity of a user or service, and authorization determines their access rights**. Although the two terms sound alike, they play separate but equally essential roles in securing applications and data. Understanding the difference is crucial. Combined, they determine the security of a system.

Common problems faced in Web testing

* Integration. Integration testing exposes problems with interfaces among different program components before deployment. ...
* Interoperability. ...
* Security. ...
* Performance. ...
* Usability. ...
* Quality Testing, Exceptional Services.

**• When to used Usablity Testing?**

Once you've got an idea, conduct usability testing **before putting any design resources to work**. Identify specific areas where testing and validation can enhance your concept. After you get the results from your initial test, share them with your team. Then, continue testing users as you build a prototype.

**• What is the procedure for GUI Testing?**

* Testing the size, position, height, width of the visual elements
* Verifying and testing the error messages are displayed or not
* Testing different sections of the display screen
* Verifying the usability of carousel arrows
* Checking the navigation elements at the top of the page
* Checking the message displayed, frequency and content
* Verifying the functionality of proper filters and ability to retrieve results.
* Checking alignment of radio buttons, drop downs
* Verifying the title of each section and their correctness
* Cross-checking the colors and its synchronization with the theme