ISTQB Foundation Level

Day 2

By Mr. Pavan

Agenda - Day 2

- Test Design Technique
- Test Management
- Tool Support for Testing
- Exercise
- Open House…

Test Design Technique

Test Design

- Act of creating and writing test suites for testing a software
- Test analysis and identifying test conditions gives us fair idea of needed test coverage

Test Analysis

- Process of looking at something that can be used to derive test information
- Basis for the tests is called the '**test basis**'
- Test basis help analyze what could be tested - these are the **test conditions**
- The test conditions that are chosen will depend on the test strategy. They might be based on:
  - risk
  - models of the system
  - likely failures
  - compliance requirements
  - expert advice or heuristics

Dynamic Testing

- Requires that the code be compiled and run
- Involves working with the software, input values are given and output values are checked with the expected output

Equivalence Partitioning

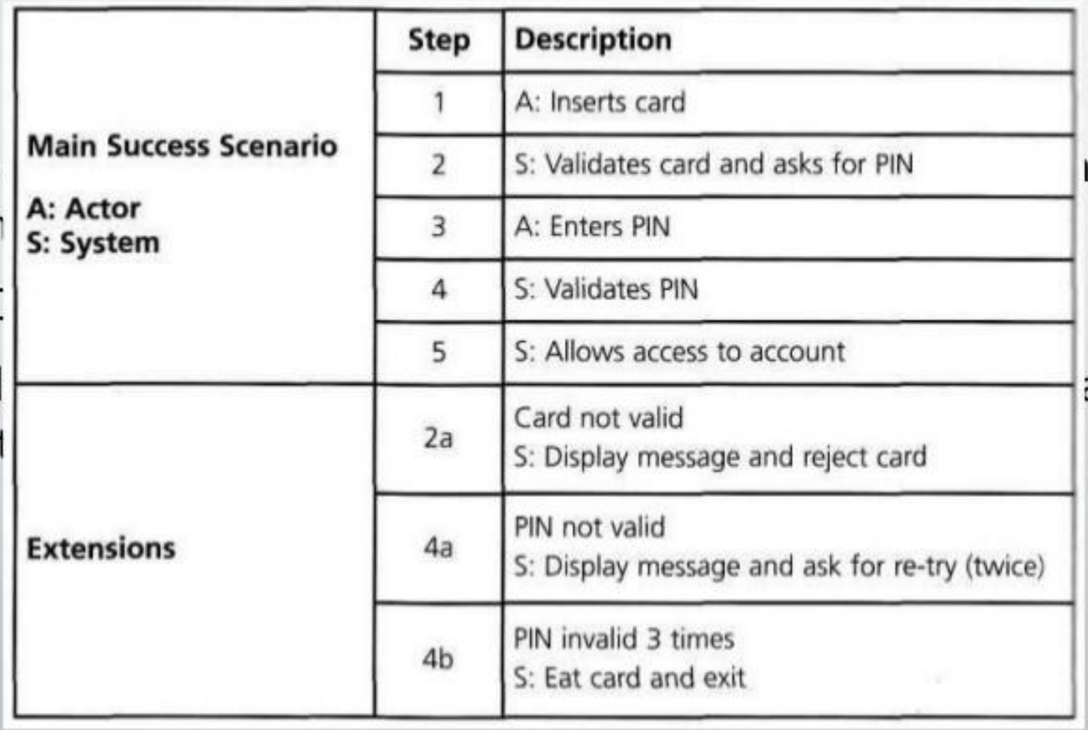
- Blackbox-Dynamic Testing technique
- Input domain data is divided into different equivalence data classes
- We need to test only one condition from each partition. This is because we are assuming that all the conditions in one partition will be treated in the same way
- **Example**: Application is accepting input range from 1 to 100
- One is for valid input class i.e. Any one value from 1-100
- One is for invalid data below lower limit i.e. any one value below 1.
- One is for invalid data above upper limit i.e. any value above 100.

Boundary value Analysis

- Identify errors at boundaries rather than finding those exist in center of input domain
- Is next part of Equivalence partitioning for designing test cases
- **Example**: Application is accepting input range from 1 to 100
- One test case for exact boundary values of input domains each means 1 and 100
- One test case for just below boundary value of input domains each means 0 and 99
- One test case for just above boundary values of input domains each means 2 and 101

Use Case Testing

- Usecase is ıd user of the system
- Blackbox-
- Helps us id ansaction by transact
- **Example**:



	Step	Description
Main Success Scenario **A: Actor** **S: System**	1	A: Inserts card
	2	S: Validates card and asks for PIN
	3	A: Enters PIN
	4	S: Validates PIN
	5	S: Allows access to account
Extensions	2a	Card not valid S: Display message and reject card
	4a	PIN not valid S: Display message and ask for re-try (twice)
	4b	PIN invalid 3 times S: Eat card and exit

Experience-based testing

- Experience of both technical and business people is required, as they bring different perspectives to the test analysis and design process
- Used for low-risk systems, but this approach may be particularly useful under extreme time pressure – in fact this is one of the factors leading to exploratory testing

Test Management

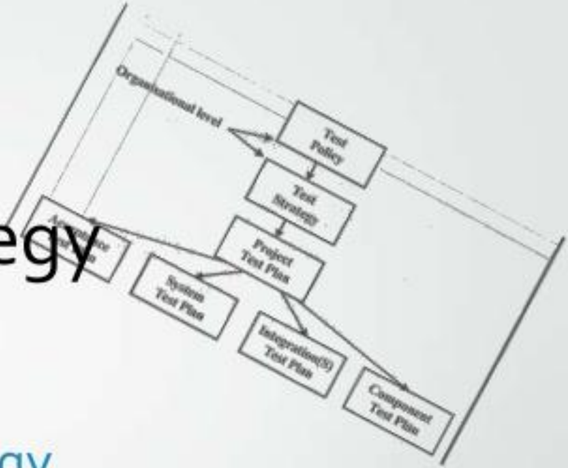
R&R – Test Leader

- Estimate the testing to be done
- Lead, guide and monitor the analysis, design, implementation and execution of the test cases, test procedures and test suites
- Ensure proper configuration management of the testware produced and traceability of the tests to the test basis
- Issue prioritization and resolution facilitation
- Identify unit testing coverage and ensure any gaps are documented and addressed
- Provide formal sign-off on all testing deliverables and events

R&R - Tester

- Reviewing and assessing requirements
- Execute and log the tests, evaluate the results and document problems found
- Setup and monitor test beds
- Review test specifications, defect reports and test results

Test Plan V/s Test Strategy



Test Plan

- Is dynamic document
- Derived from function requirement document or Use Cases
- Subset of Test Strategy
- Contents - Introduction, Test items, Features to be tested, Features not to be tested, Test techniques, Testing tasks, Suspension criteria, Features pass or fail criteria, Test environment (Entry criteria, Exit criteria), Test delivarables, Staff and training needs, Responsibilities, Schedule

Test Strategy

- Is a static document
- Derived from BRS
- Superset of Test Plan
- Contents - Scope and Objectives, Roles and responsibilities, Communication and status reporting, Test deliverability, Industry standards to follow, Testing measurements and metrics, Risks and mitigation, Defect reporting and tracking, Change and configuration management, Training plan

Estimation Technique

- **Estimation** is process of predicting the most realistic use of effort required to develop or maintain software
- Technique used to calculate the time required to accomplish a particular task is **Estimation Techniques**
- Various estimation techniques are:
  - **Function Point Analysis**
  - **Work Break Down Structure**
  - **Delphi Technique**

Function Point Analysis

- Estimates can be by one who understand the system from a functional perspective
- Analysis of functional user requirements of the Software with following categories:
  - Outputs
  - Inquiries
  - Inputs
  - Internal files
  - External files

Work Breakdown Structure

- Involves breaking requirements down into individual components in a hierarchical structure
- Provides traceability of task completion
- Estimation happens at most granular level

Delphi Technique

- Based on surveys and basically collects the information from experts
- Requirements are estimated in two or more rounds
- Mean or median is usually taken as final

Configuration Management

- Management of source code, test scripts, third-party software, hardware, data etc.
- Supports the build process, which is important for delivery of a test release into the test environment
- Allows us to keep the record of what is being tested to the underlying files and components that make it up. Let us take an example, when we report defects, we need to report them *against* something, something which is **version controlled.**

Risk Based Testing

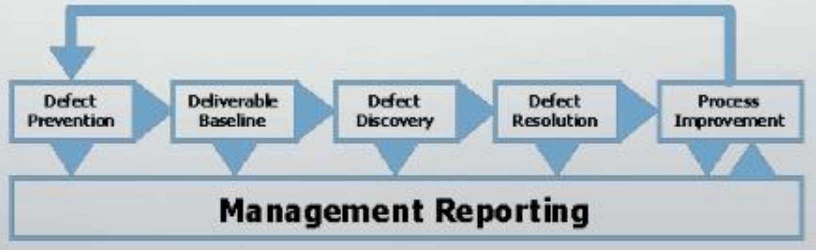
- Uses risk to prioritize and emphasize the appropriate tests during test execution
- Usually done when there might not be sufficient time to test all functionality
- Involves both mitigation – testing to provide opportunities to reduce the likelihood of defects
- Starts early in the project, identifying risks to system quality and using that knowledge of risk to guide testing planning

Exploratory Testing

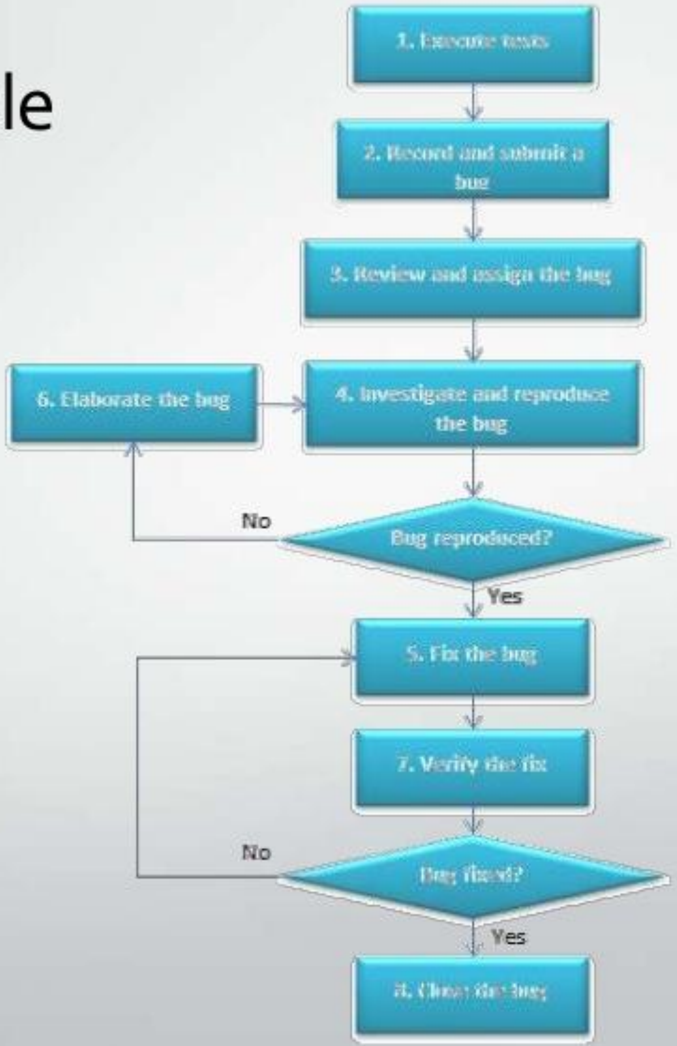
- Judgment based testing of system
- Done when there are no or poorly requirements defined
- Is a hands-on approach in which testers are involved in minimum planning and maximum test execution
- Test execution and Test design happens in parallel

Defect Management

- Process of recognizing, investigating, taking action and disposing of defects
- Helps team keep track of outstanding defects in their project effectively
- Typically it involves



Defect Lifecycle



Severity V/S Priority

- **Severity** is the extent to which the defect can affect the software
  - Is related to technical aspect of the product. It reflects on how bad the bug is for the system
  - **Show Stopper**: 4 – Cannot able to test application further.
    Major Defect: 3 – Major functionality not working but able to test application.
    Minor Defect: 2 –Bug in Functionality but in the sub module or one under the other module.
    Cosmetic: 1 – Issues in location of the object or the look and feel issue
- **Priority is** order in which we should resolve a defect
  - Based on the customer requirements

    Low: Repair can be deferred until after more serious defect have been fixed.
    Medium: Should be resolved in the normal course of development activities. It can wait until a new build or version is created.

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

Examples

- **High Priority & High Severity:** Upon login to system “Run time error” displayed on the page
- **Low Priority & High Severity:** On the home page of the company’s web site spelling mistake in the name of the company
- **Low Priority & Low Severity:** Spelling mistake in the confirmation error message

Tool Support for Testing

Types of software testing tools

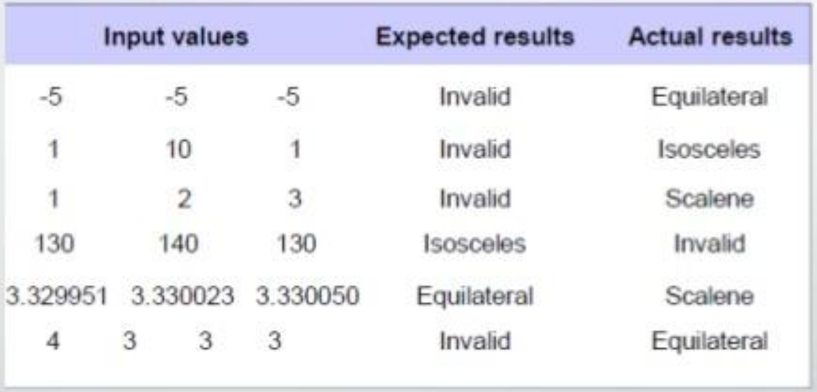
- Tools are grouped by the testing activities or areas that are supported by a set of tools
- Test Management tool may provide support for managing testing (progress monitoring), configuration management of testware, incident management, and requirements management and traceability
- Defect Management Tool tracks details, assignment, action, and reporting of defect induced in various categories

Exercise

Triangle Problem

- **Problem** -> Read in three numbers representing the lengths of the sides of a triangle, and to print out a description of what kind of triangle it is
- **Test Basis** ->
  - Equilateral (if all three sides have equal length)
  - Isosceles (if two sides have equal length)
  - Right-angled (if one angle is a right angle, excluding isosceles)
  - Scalene (all sides different lengths, excluding right angled)
  - Impossible (if the lengths can't form a triangle, or give zero area)
  - Invalid (if the input is not in the required form)

Triangle Problem



Input values				Expected results	Actual results
-5	-5	-5		Invalid	Equilateral
1	10	1		Invalid	Isosceles
1	2	3		Invalid	Scalene
130	140	130		Isosceles	Invalid
3.329951	3.330023	3.330050		Equilateral	Scalene
4	3	3	3	Invalid	Equilateral

Click Here to Submit Feedback