Practical Software Testing - eBook

Prepared by Chindam Damodar
Published by http://SoftwareTestingHelp.com

Dedicated to my

Mother Smt.Chindam Rajeshwari and Father Chindam Rajamouli

and

To my friend Vijay & all the STH readers.

PREFACE

It is pleasure to present this second version of Software Testing e-book to SoftwareTestingHelp.com readers. Like the first edition, I think this improvised and updated version would also fetch more admirers. There are significant changes in the Software Testing pertaining to these recent days.

It was an honor to share with all of you the first edition of the Software Testing e-book which received an overwhelming response. The feedback and reviews received have been very encouraging. We share the joy of our debutante success with you for your valuable feedback and moral support in our dream endeavor. I would like to specially thank my brother Ch.Naresh and my parents, for their tremendous cooperation and encouragement in the course of producing this work.

Software Testing

Foundation Program in Software Engineering

Introduction:

Definitions of Software Testing:

Testing*: *Testing an application to detect differences between existing and the required condition and evaluate the features of the software application.

Manual Testing*: *Testing activities performed by the people (Testers) without the help of any Software Testing Tools.

*** Testing is based on user requirements. This is done in order to find out any defects that might be the cause of the program or system to fail in meeting the client requirements.

Fundamentals of software testing:

Software Application: A set of Computer program and minimal data use to run a system is called software.

Example: 1) Accounting Software
2) Emailing Software and others

Software application is basically categorize in- two types:

1) Projects 2) Products

1) **Projects:** If a software application is developed for a specific customer requirement then it is called a project.

2) **Products:** If a software application developed for multiple customer requirements then it is called a product.

Errors: Any incorrect human action that produces a problem in the system is called an error.
Defect: Deviation from the expected behavior to the actual behavior of the system is called defect.

Failure: The deviation identified by end-user while using the system is called a failure.

Software Development Life Cycle (SDLC)

- SDLC is a process of developing various information systems.
- Phases in SDLC:

1) Requirement Capturing
2) Analysis
3) Design
4) Coding
5) Testing
6) Implementation
7) Support / Maintenance.

Request for Proposal (RFP):

- Sony will prepare and distribute the RFP to selected vendors (Software Companies).
- Software companies respond to the RFP with their proposal (Information about the company, Technical solution, Delivery Schedule, Budget, Time Line and deliverable etc.)
- If Sony likes the proposal of any vendor, then it will sign the Statement of Work (SOW) with that vendor.
- Then a project kick off meeting (it's a high level meeting between the Senior Manager, CTO and Project Managers) is conducted.
- Identify the Project Manager.
- The Project Manager will prepare Project Management Plan (PMP).
- After a high level meeting with the customer Business team/Technical team will release Business Requirement Specification (BRS).
- BRS is a high level document containing project requirement from customer.

- Person should be domain expert/functional expert is eligible to convert the BRS into SRS (Software Requirement Specification)

QA and Testing:

- Identify Standards, Guidelines, Procedures, Checklist and Templates required for the project.
- Review and Approval of SRS (Mapping SRS with BRS)
- Once SRS is approved create a baseline to SRS (SRS 1.0V)

Development:

- Based on the approval, the SRS Software Architecture will design the model of the project (Blue Print / Prototype)

Model Contains:

1) Architectural Design
2) Database Design
3) User Interface Design
4) All UML Diagram
5) Use Case
6) Class
7) Object
8) Sequence
9) Collaboration
10) 9.1) Activity
11) Deployment…………

- Review and approval of design document and mapping it with the design documents.

Q & A Testing:

- Test Manager/Test Lead will prepare a test plan for the project
- Review and Approval of test plan
- Test Engineer will identify the Test Scenarios of a project (What is to be Tested)
- Review and Approval of Test Scenario’s

- Test Engineer will prepare Test Cases of all approved Test Scenario's (How/what is to be tested).
- Review and approval of test cases

Development:

- Based on the approved design documents, developers convert the logic specified in the design documents into coding as per the chosen programming language and then they will release code (LOC)
- Perform code reviews
- Once code is approved, perform unit level testing
- Once ULT is approved; developer will integrate the units and release the build for testing.

Q & A Testing:

- Perform integration level testing
- Perform Retest and Regression testing
- Once integration level testing is approved, perform system level testing.
- Again perform Retesting & Regression testing.

Development:

- Developers are going to fix the defects which are reported by testing department.

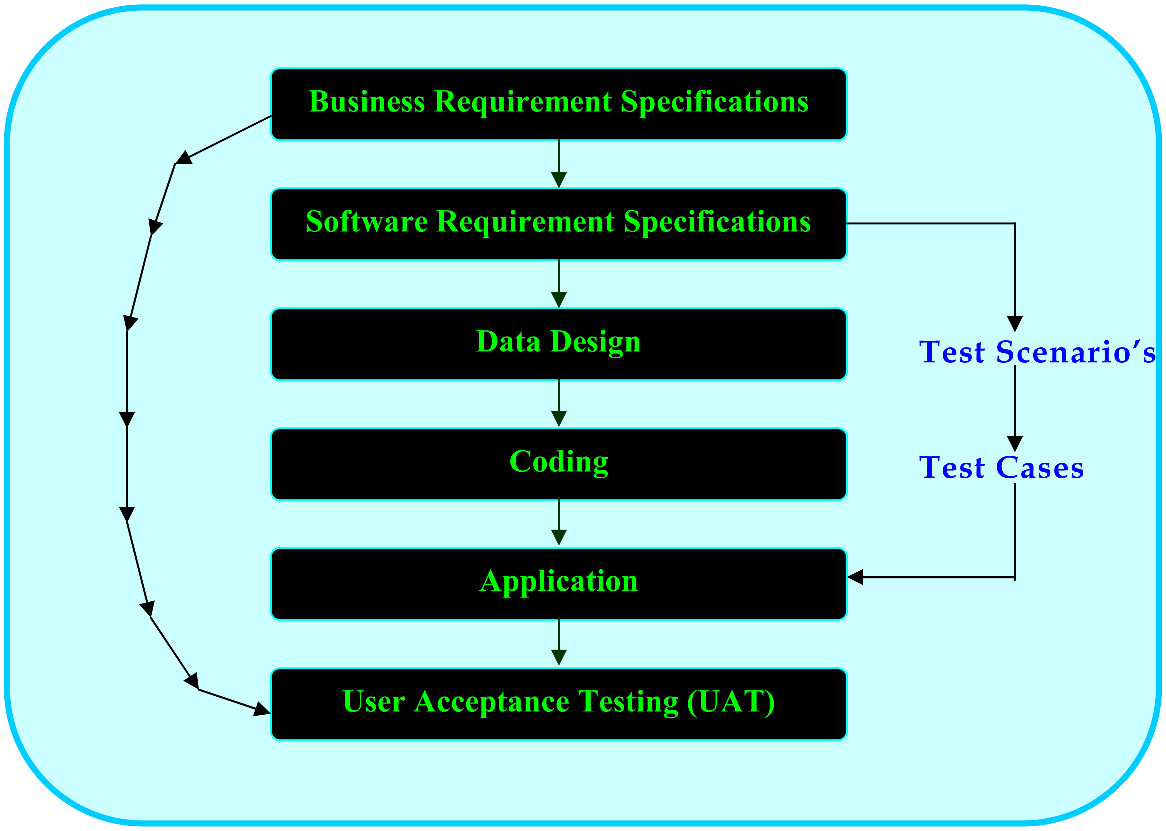
Q & A Testing:

- Once System Level Testing is approved, release the application for User Acceptance Testing (UAT).
- Perform U.A.T.
- Once U.A.T. is successfully performed, release the application for product department to create exe.
- Once exe is ready perform installation testing.

- Once installation testing is approved implement the developed software into customer environment and provide user training.
- Provide support to the customer for defects questions issues future enhancements.

***Note:**

Presence of errors results in defects and presence of defects results in failure of the product.


Business Requirement Specifications
Software Requirement Specifications
Data Design
Coding
Application
User Acceptance Testing (UAT)
Test Scenario's
Test Cases

** When defects will arise while developing software.

Request 1	**Request 2**	**Request 3**	**Request 4**
Correct Requirement	Correct Requirement	Correct Requirement	Incorrect Requirement
Designed as per requirement	Designed as per requirement	Mistakes made in design	Designed as per requirement

Developed as per design	Mistakes in development	Developed as per design	developed as per design
Correct Product	Product has coding defects	Product has design defects	Wrong product.

* The above diagram explains the importance of early testing.

Early Testing:

Conducting testing as soon as possible in development life cycle helps in finding defects at an early stage and this is known as Early Testing. Early testing is helpful to reduce the cost of fixing defects.

*** Why there are defects in software?**

* Incorrect requirements
* Wrong design
* Poor coding
* Complex business logic
* Complex technology
* Work pressure
* Frequently changing requirements.

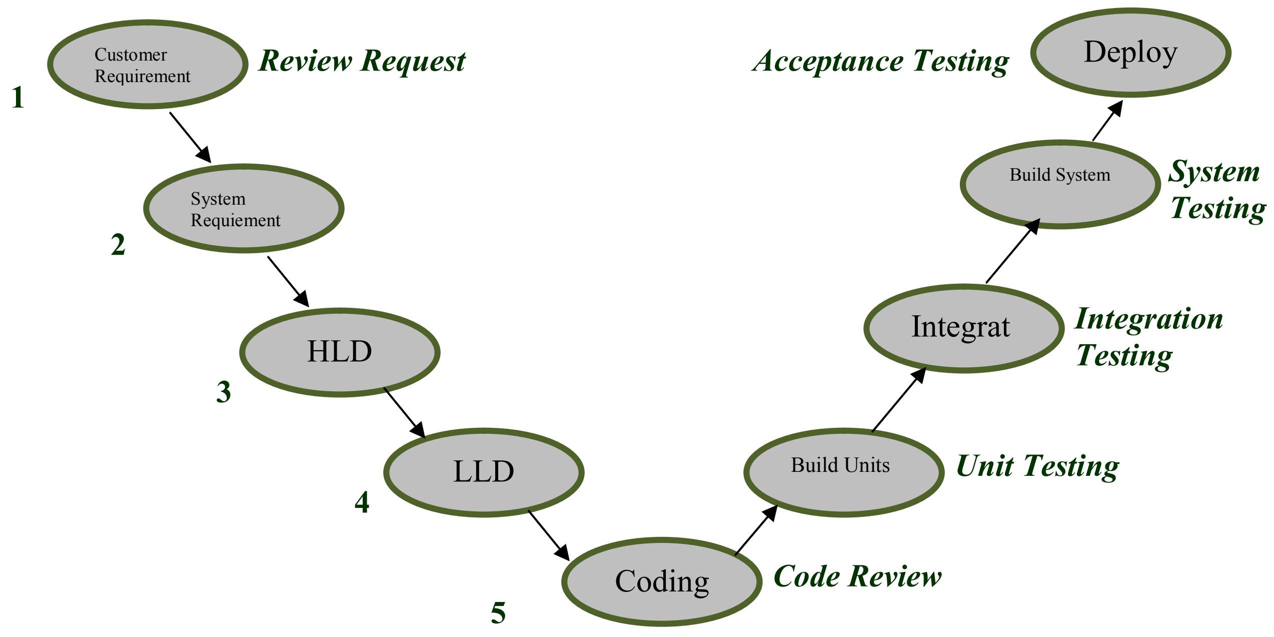
Testing: It is a process of verifying if we are developing the right product or not and validating if the developed product is right or not.

Software testing = Verification + Validation.

*** What is Verification?**
Are we building the system right? It's a process of checking if the system is well-engineered. It is also called static testing.

*** What is Validation?**
Are we building the right system? It is a process of checking if system meets the customer's actual requirements. It is also called dynamic testing.

Verification Vs Validation



***** 1, 2,3,4,5, are Verification and other side is Validation.

Software Testing Techniques:

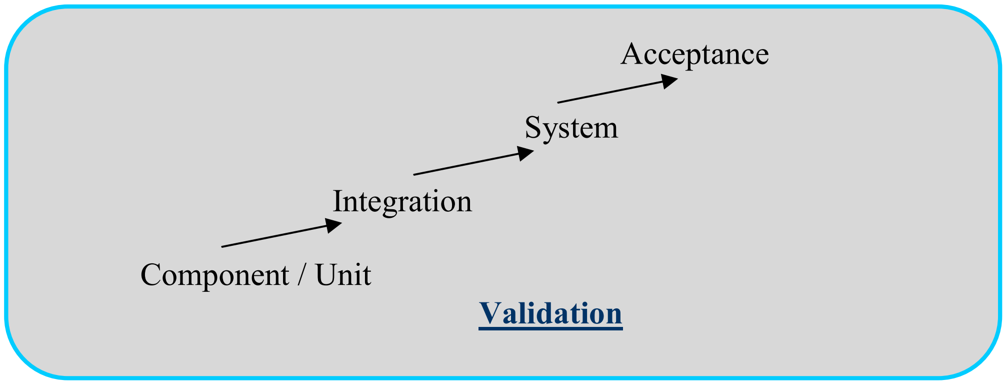
* Static Testing

* White Box Testing

* Black Box Testing

* Grey Box Testing.

Levels of Dynamic Testing:



SDLC Models

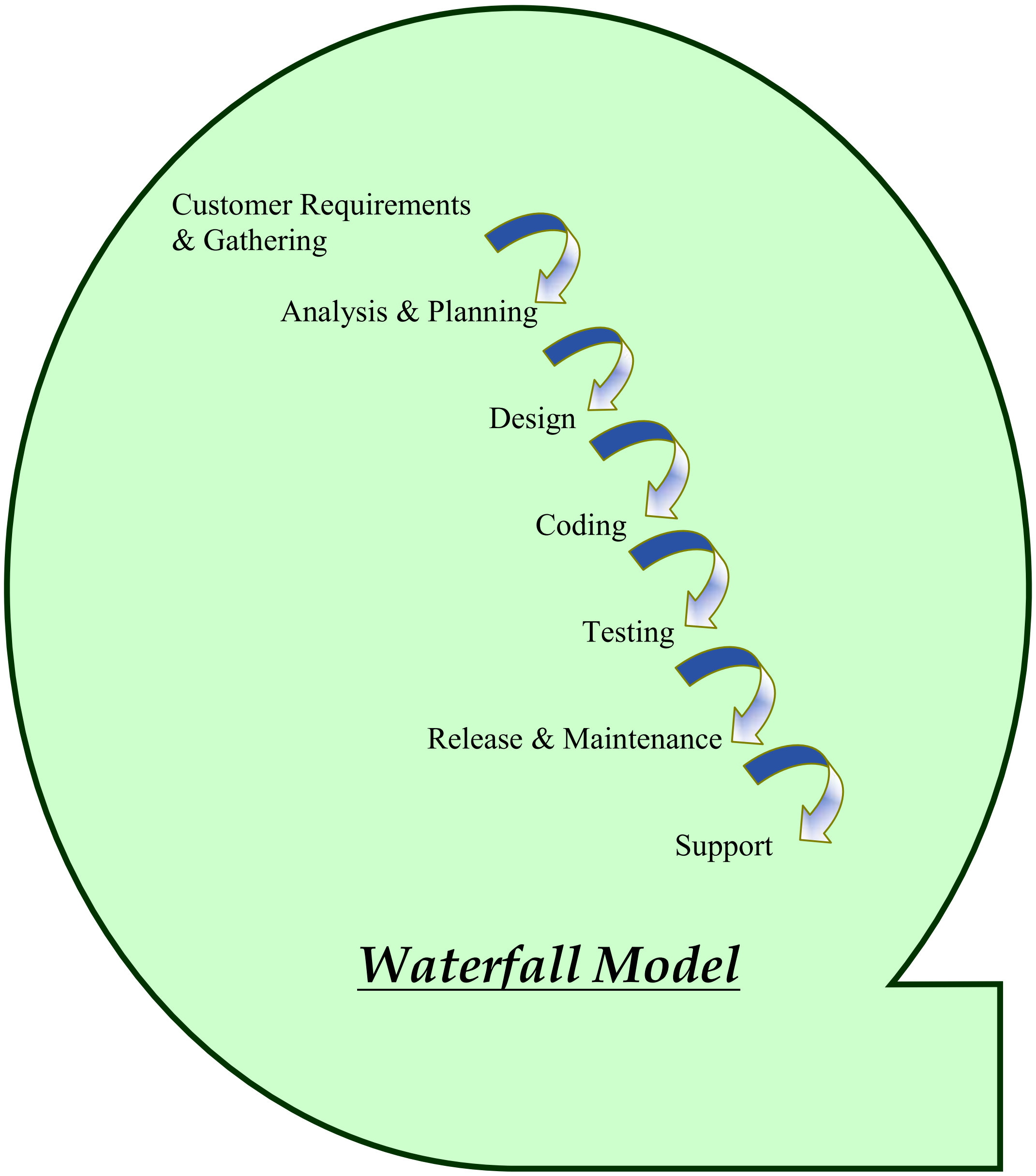
What is SDLC Model?

A framework containing the process activities and the tasks involved in the developments, operations and maintenance of a software project spanning the life of the system from the definition of its requirements to the termination of its use.

Types of SDLC Models:

1) Water fall Model
2) Spiral Model
3) Prototype Model
4) Rapid Application development Model (RAD)
5) Iterative and Incremental development Model
6) Agile Model.

1) Waterfall Model:



Customer Requirements & Gathering →Analysis & Planning →Design (Low Level –High Level) →Coding →Testing →Release & Maintenance →Support.

- It is also called as "**Classic Life Cycle Model**", **Linear Sequential Model**.
- This model suggests systematic and sequential approach to software development that begins at requirements and gathering and progress through all the phases of the life cycle sequentially.
- Development activities carried out sequentially.

- Review and Approval of each phase outputs.
- Model does not permit going back and forth.
- If any defect is found, you can revert to the originating phase and start traversing sequentially all over again.

***Suitable for Project where:**

- Requirements are clearly defined
- Small and Medium term duration
- Use of Stable technology
- Familiarity with the domain and development environment.

Advantages:

- Project under control
- Predefined outputs at every phase.
- Tracking changes is easy
- Early identification of slippages, if any

Dis-Advantages:

- Customer requirement may change
- Customer appraisal of completed work – not feasible always.
- Phases cannot run concurrently.

2) Spiral Model:

- In spiral model, software is developed in a series of 'incremental' releases
- The spiral model is divided into a number of framework activities or task.
- Suitable for large projects with multi-location implementation
- Each spiral consists of a deliverable product.
- Feed back of each spiral is incorporated in the text spiral.
- Customer can start using the system after every spiral
- Each spiral consists of waterfall model.

Advantages:

- Useful for large projects
- Customer requirements evolve over a period
- Early availability of usable system.

3) Prototype Model:

- Identify - It identifies the basic requirement
- Initial prototype – A prototype is developed based on the initial understanding of the customer requirements
- The initial prototype developed consists of the user interfaces only.
- Review - The customers, including end-users, examine the prototype and prototype feedback on additions or changes.
- Revise and Enhance the prototype:
    - Using the feedback both the specifications and the prototype can be improved
- "A visible working prototype helps customer to define the requirements"

Advantages:

- It can be used when customer is not sure about what he wants.
- It is a faster way of finalizing the requirements
- It is useful for new technologies and domains.

4) Rapid Application Development (RAD)Model:

- RAD enables creation of fully functional software within a very short time.
- If the requirements are well defined and understood, and the project scope is constraint, the RAD process enables a development team to create a fully functional system within a very short time period.

5) Iterative and Incremental development Model:

- The basic idea behind iterative enhancement is to develop software system incrementally.
- At each iteration, design modifications are made along with adding new functional capabilities.
- In an incremental development, the system is developed in different stages, with each stage consisting of requirements, design, development, and test phases; in each stage a new functionality is added.
- This type of development allows the user to see a functional product very quickly and allows the user to impact what changes are included in subsequent releases.

6) Agile Model:

- In Agile Model project is divided into various sprints
- Each sprint contains High-Priority Requirements.
- The time period for sprint is typically 2-4weeks.
- In an Agile model, daily screen meetings with team to share status and potential issues.
- Each sprint is released to customers.
- Used for critical applications.

Software Configuration Management (SCM)

- The process of identifying, organizing and controlling changes to the software during the development and maintenance phase.
- A methodology to control and manage a software development project.

Purpose of SCM:

Establish and maintain the integrity of work products.

SCM Needs:

- Multiple people have to work on software that is changing.
- Projects delivering several releases (Builds)
- Software must run on different machines and operating systems (Rapid evolution of software and hardware)

Problems resulting from poor Configuration Management:

- Can't roll back to previous subsystem
- One changed overwrites another
- Which code change belongs to which version.
- Faults which were fixed Re-appear.
- Tests worked perfectly on old versions.

Configuration Management Using:

- Multiple project members – concurrent access
- Orderly control of change
- View difference
- Rollback changes
- Release Management
- History
- One of the key process area in SEI CMMI Level2.

Major Activities:

- Configuration planning & Step
- Configuration identification
- Configuration baseline
- Change Management
- Configuration release control
- Configuration Audit
- Control of customer property.

Sample list of Software configuration items:

- Management plans (Project Plan, Test Plan, etc.,)
- Specification (Requirements, Design, Test cases, etc.,)
- Customer documentation(Implementation, Manuals, User Manuals, Online help files)
- Source code(Java, .Net, PHP, VB, etc.,)
- Executable code (exe's)
- Libraries (Packages, %includes.files, API'S, DLL'S, etc.)
- Databases (Data being processed, testdata, etc.)
- Production documentation.

(Visual Source Safe) MS VSS6.0 Navigation:

- It is commercial software from Microsoft.
- Available in two types: i.e.,

  1) Server, 2) Client

* Test Engineer responsibilities in configuration management i.e., VSS6.0

1) Copy files from VSS i.e., checkout files from repository
2) Check in files after completion of your work.

********* Current version we worked on Microsoft Visual Source Safe 6.0**

Software Test Engineer Responsibilities:

- Understanding the requirements and functional specifications of the application
- Identifying the required test scenario's for the project
- Designing and preparing Test cases to validate the application.
- Execute Test cases to valid the application
- Log Test results (How many test cases pass or fail)
- Defect reporting and tracking
- Retest fixed defects of previous build
- Performed various type of testing assigned by Test Lead (Functionality, Usability, User Interface and compatibility) etc.,
- Reports to Test Lead about status of the assigned tasks
  - Daily status report
  - Daily defect report
  - Weekly status report
  - Retesting Report
  - Other assigned tasks if any
- Participated in regular team meetings by lead and manager
- Creating enhancements

Testing Process / Life Cycle:

- Understanding the requirements and functional specifications of the application.
- Identifying required Test Scenario's for Project.
- Designing and preparing Test Cases to validate application.
- Execute Test Cases to valid application
- Log Test results (How many test cases pass/fail)
- Defect reporting and tracking.
- Retest fixed defects of previous build
- Perform various type of testing assigned by Test Lead (Functionality, Usability, User Interface and compatibility) etc.,
- Reports to Test Lead about the status of assigned tasks
  - Daily Status Report
  - Daily Defect Report
  - Weekly Status Report
  - Retesting Report
  - Other assigned tasks any
- Participated in regular team meetings by lead and manager
- Creating automation scripts for Regression Testing.
- Provide enhancements to project,-from the End user's perspective.
- Provides recommendation on whether or not the application / system is ready for production.

Real Time Manual Testing

*** Project Management Plan (PMP):**

The Project Manager creates the project plan for project.

*** Entry Criteria to Prepare PMP:**

- RFP
- Sow (or) Work Order
- Project Plan Template

Exit Criteria to Prepare PMP:

- Project should be Reviewed and Approved.
- Once PMP is approved, configuration controller will create baseline for pmp (PMP 1.0Version) and will update the same in the configuration repository. (File Server or VSS).

Main Aspects of PMP:

- Scope Management
- Schedule Management
- Quality Management
- Resource Management for resources like people, tools and others.
- Communications Management
- Risk Management
- Data Management

Requirement Specification

Types of requirements

- Business requirement specification
- Software requirement specification
- Functional requirement specification

*** Business Requirement Specification:**

- It describes in professional terms the requirements of the client and what needs to be delivered to client.
- It's a high level document, containing project requirements from customer.
- It contains the problems of the existing system which the client is facing. A Person who is a domain expert or functional expert is mostly able to convert the BRS into SRS.

*** Software Requirement Specification (SRS):**

- An SRS is basically an organization's understanding of a customer or client's systems and dependencies at a particular point in time prior to any actual design or development work.
- It's a two way insurance policy that assures that both the client and the organization understand each other's requirements.

*** Functional Requirement Specification (FRS):**

- A Functional Specification document describes how a project will work entirely from the user's perspective. It describes the expected features and specifies details for screens, menu, dialogs, etc.

- A Functional Specification document defines a software system or its component. A function is described as a set of inputs the behaviour and outputs.
- A FRS defines how the system must behave when presented specific inputs or conditions. These may include calculations, data manipulations, processing and other specific functionality.

Test Plan

Test Manager (or) Test Lead (or) Test Engineer will study the approved SRS and prepare a Test Plan for project based on the approved PMP.

*** Entry Criteria to Prepare Test Plan:**

- Approved PMP

- Approved SRS

- Test Plan Template

*** Exit Criteria to Preparing for Test Plan :**

- Test Plan should be reviewed and approved.

- Once Test Plan is approved. Test Lead will create baseline for the Test Plan

(Test Plan 1.0V) and will update same in the configuration repository

(File Server (or) VSS)

*** Definition:**

1. A document describing the scope, approach, resources and schedule of testing activities. It identifies test items, the feature to be tested, the testing tasks, who will do each task, and any risks requiring contingency planning.

2. A detail of how the test will proceed, who will do the testing, what will be tested, in how much time the test will take place, and to what quality level the test will be performed.

*** Table content of Test Plan:**

1. Introduction
    - 1.1 Test Plan Objectives
2. Scope of this document
3. Test Strategy
    - 3.1 Smoke Testing
    - 3.2 Sanity Testing
    - 3.3 System Testing
    - 3.4 Database Testing
    - 3.5 Cross browser Testing
    - 3.6 Automation Testing.
4. Environment requirements
5. Test Schedule
6. Control Procedures
    - 6.1 Reviews
    - 6.2 Bug Reviews
    - 6.3 Change Request
7. Functional to be Tested
8. Functions not be Tested
9. Resources and Responsibilities
10. Deliverables and Milestones
11. Defect Management
12. Dependencies.
    - 12.1 Personal dependencies
    - 12.2 Software dependencies
    - 12.3 Hardware dependencies
    - 12.4 Test data & database
13. Risks ------Known (or) Unknown
14. Tools

15. Documentation
16. Approvals
17. Entry / Exit for each Testing activity
18. Test Suspension
19. Test Resumption
20. Test Completion

Test Scenario's

* Identify all the possible areas to be tested

(or)

* What is to be tested.

* **Entry Criteria to Identify Test Scenario's:**
  - Approved Test Plan
  - Approved SRS
  - Any design document like UI etc.
  - Test Scenario template

* **Exit Criteria for Identifying Test Scenario's:**
  - Test Scenario's should be reviewed & approved
  - Mapping Test Scenario's against requirements
  - Once Test Scenario's are approved TL will create the baseline for Scenario's (TS1.0V)
  - And he will update the same in the configuration repository.

Test Scenario Template

Project Code : CORE_111	**Reviewed by:**
Project Name: WAPR	**Date Reviewed by:**
Identified by : Tester1	**Approved by:**
Date Identified: 18.09.2012	**Date Approved:**

TS#	Req #	Main Functionality	Sub Functionality	Test Case Name	Test Case Objective	Referer/ Doc-ID
TS_001		WAPR	Home Page	WAPR_Home Page	Verify the usability and User interface of Homepage	UID_Page1
TS_002		WAPR	Home Page	WAPR_Home Page	Verify the functionality of 'Home'	
TS_003		WAPR	Home Page	WAPR_Home Page	Verify the functionality of 'About Us'	
TS_004		WAPR	Home Page	WAPR_Home Page	Verify the functionality of 'Contact Us'	
TS_005		WAPR	Home Page	WAPR_Home Page	Verify the functionality of 'Product info'	
TS_006		WAPR	Home Page	WAPR_Home Page	Verify the functionality of 'Help'	
TS_007		WAPR	Home Page	WAPR_Home Page	Verify the functionality of [Submit]	
TS_008		WAPR	Home Page	WAPR_Home Page	Verify the functionality of [Reset]	
TS_009		WAPR	Home Page	WAPR_Home Page	Verify the functionality of [Close]	

Test Case Template

						Failed						
		Test Case History				**Not Executable**						
		Created By	Tester1	**Date Created**	18.09.12	**Defects Reported**		**Test Case Summary**				
		Reviewed By	Tester2	**Date Reviewed**	20.09.12			Total No. of TCS				
		Approved By	TestLead	**Date Approved**	20.09.12			Passed				
								Failed				
		Test Information						**Not Executble**				
		Test Case Name	WAPR Home Page	**Test Case Objective**	Verify the Usability, Userinterface and functionality of Home Page			Defect Reported				
		Test Executed By		**Date Executed**								
		Version		**Build**								
TC #	**TS#**	**Test Design/ Steps**	**Input Data**	**Expected Result**	**Actual Result**	**Pass**	**Fail**	**Comments**	**Critical**		**Severity Major Minimal**	**Moderate**
1		Launch the browser, Enter the valid URL and click on [GO] or Press [Enter]	www.wapr.com	Should display the WAPR Home Page								
2		Enter valid Customer ID and password Click on [Submit]		Should display the Customer Home Page								
3		Verify Spellings, Fonts, Alignments of Customer Home Page		Should not be any Spelling Mistakes, Should be a same Fonts, same Alignment								
4		Focus on 'Logout'and Press tab Continuously		Tab order should go wih all the fields according totheir occurrences								
5		Click on 'View Profile'		Should display a View Profile Page								
6		Clicck on 'Edit Profile'		Should display a Edit Profile Page								
7		Click on 'Change Password'		Should display a Change Password Page								
8		Click on 'Shop Now'		Should display a Shop Now Page								
9		Click on 'Logout'		Should display the WAPR Home Page								
10		Close the Browser		Browser should be closed								

Test Case

A test case is a document which contains a set of input values, execution preconditions, expected result which is developed to achieve a particular objective, such as to verify compliance with specific requirements.

(or)

How "What is to be tested".

*** Entry Criteria to Prepare Test Case:**

- Approved Test Plan
- Approved SRS
- Functional requirement specification
- Approved Test Scenario's
- Test Case Template
- Test Case Guidelines

*** Exit Criteria for Preparing Test Case:**

- Test Case should be reviewed and approved
- Mapping Test Cases (TC#) against Test Scenario's (TS#) and Requirement (Req#)
- Once the Test Case approved the TL can create the baseline.
- And also will update the same in the configuration repository.

*** Test Case contains (or) Test Case Template contains:**

- A test case should contain particulars such as test case name, objective, Test conditions, input data requirements, expected results and the name of the person who prepared it

* What is a testing technique?

- A procedure for selecting or designing tests based on a structure or functional model of the software.
- Successful at finding faults
- A best practice
- A way of delivering good test cases.
- A way of objectively measuring a test effort.

* Test Case design technique:

They are 2 categories

1) Black Box (Functional) Test case design techniques

2) White Box (Structural) Test case design techniques

1) Black Box (Functional)Test Case Design Techniques:

- Equivalence Class Partitioning
- Boundary Value Analysis
- State transitions testing
- Cause effect graphing

*)Equivalence Class Partitioning (ECP):

- ECP is a method for deriving test cases.
- It identifies similar functionalities and divides equally
- Labels the classes as "valid" and "invalid"

Example:

If an item can be -9999 to +9999 identify the equivalence classes?

Valid	-9999 to +9999
Invalid	-10,000 to 10,000

Less than -9999, Greater than +9999

Another Example:

* Name [ ] 5 to 30 Alphabets

Here Blank is invalid

$$\text{BVA}\begin{cases} \left.\begin{array}{l}4\\31\end{array}\right\}\ \text{Valid Boundary}\\ \left.\begin{array}{l}5\\30\end{array}\right\}\ \text{Invalid Boundary}\end{cases}$$

$$\text{ECP}\begin{cases} \text{5 to 30 ------Valid Partition}\\ \left.\begin{array}{l}< 5\\> 30\end{array}\right\}\ \text{Invalid Partition}\end{cases}$$

Project Testing

- Once all the WebPages are integrated at one level by developers they will release the build for testing by URL's note
- The development lead (or) Test Lead will update the build and release the note into testing server and he will send an email to the test lead saying that
- Build is ready for testing
- Build URL'S

Defect:

The Problem which is identified by the Test Engineer Machine is called a DEFECT or a BUG.

Error:

The same problem which is identified by a developer machine is called an ERROR.

Failure:

A deviation from the specified or expected behaviour that is visible to the end-user, is called a FAILURE.

Testing Approach:

Traditional testing approach: Positive Testing (+ve).

- Shows that the system:
  - Does what is should
  - Doesn't do what it shouldn't.
  - **Goal:** show working
  - **Success**: system works
  - Easy to write system cases
  - **Result**: faults left in

Better Testing Approach / -ve Testing:

- Shows that the system:
  - Doesn't do what it shouldn't.
  - Does what is should
  - **Goal**: find faults
  - **Success**: system faults
  - Difficult to write test cases
  - **Result**: fever faults left in

Most common defects

- Incorrect functionality
- Incorrect data edits
- Poor performance
- Poor security
- Incompatibility
- Poor UI – User interface

- Poor usability

Objectives of Testing:

- To conform whether the application developed is according to the customer requirements or not
- Finding defects
- To make sure that the product is error free and all problems are resolved and closed
- Finally testing is helpful to deliver a quality and risk-free product to the customer.

Severity and Priority:

Severity: Severity defines the importance of defect with response to functional point of view which means criticality of the defect with respective to the application.

Responsibility of 'Testing Team '

Classification could be:
--Critical
--Major
--Moderate
--Minimal

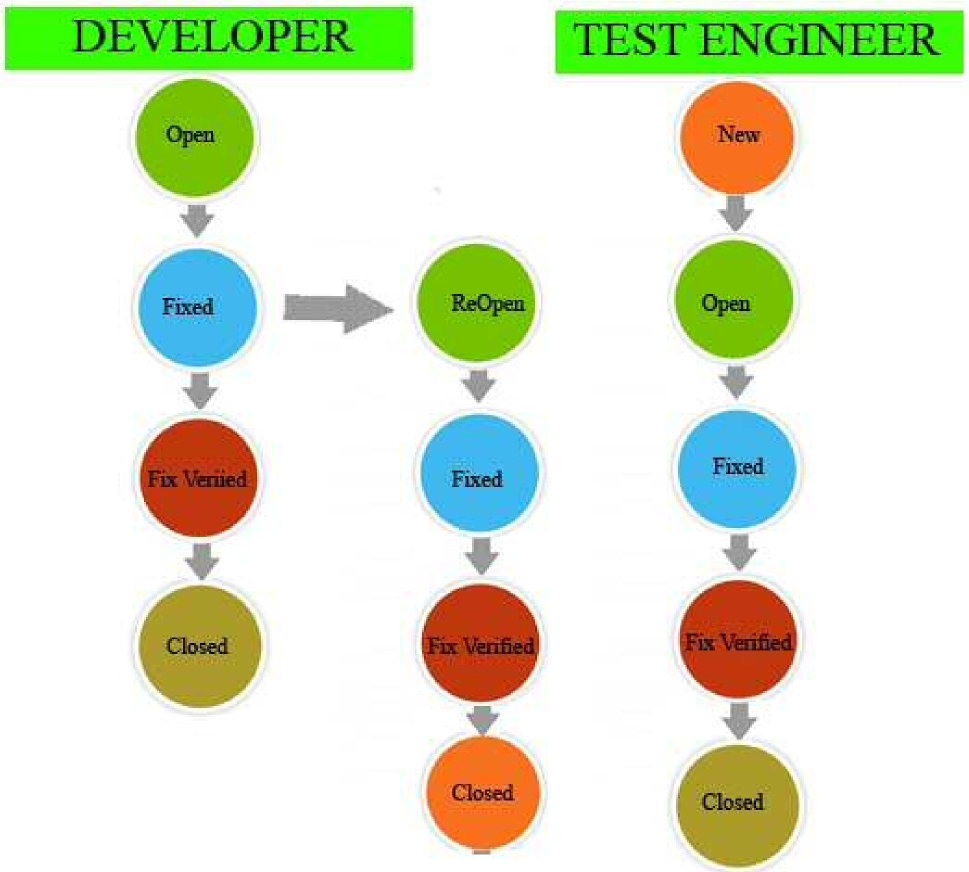
Priority: Priority defines the importance of defect with respect to client point of view which means how soon it should be fixed.

Responsibility of 'developing team'

Classification could be:

-- P1------Urgent
-- P2------High
-- P3------Medium
-- P4------Low

Defect / Bug Life Cycle



Defect / Bug Life Cycle

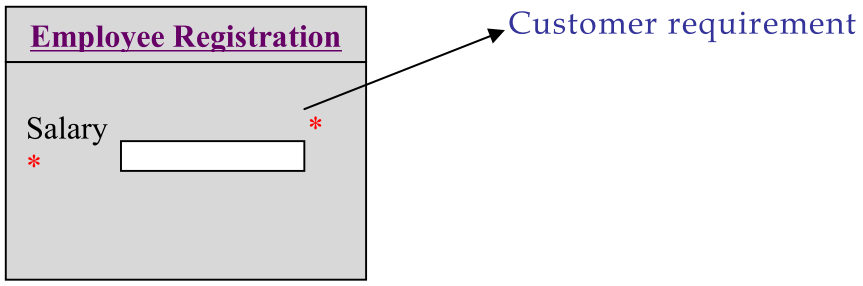
1) If the Test Engineer identifies a bug in the project, Test Engineer should report it using Bugzilla by selecting the status, severity, found in version/build and providing the steps with screen shots related to the bug.
2) Test Engineer will report the bug to the Tech Lead (Development Lead) in company at the reporting time i.e., **NEW**
3) Once the bug is **OPEN, the** developer will fix the bug. Test Engineer will be notified about the **FIXED BUG** which can be retested against a new build.
4) If the bug is fixed a **RE-TEST** is performed and the status is changed to **FIXED VERIFIED**.
5) Finally the Test Lead will verify once again and will **CLOSE** the Bug.

Software testing Principles:

Principles of testing: Exhaustive principle is impossible.

Exhaustive Testing: If you test the functionality with all possible valid inputs and invalid inputs it is called exhaustive testing.

*** Testing everything is called an exhaustive testing.



It should accept a value between 5,000-5,0000

- If you check the salary field 5000, 5001, 5002 ………….50000 and 4999, 4998 etc. it is called exhaustive testing.
- As exhaustive testing is an impossible risk based testing which is preferred (or) recommended.

Risk Based Testing: Identifying the operations which most likely can cause failures than testing these functionalities as a priority based testing is also called risk based testing.

Defect Clustering: A small number of modules or functionalities may contain more number of defects and it will require exhaustive testing on these functionalities.

Pesticide Paradox: If prepared test cases are not able to find defects, add/revise the test cases to find more defects.

The prepared test cases are not helping to find defects, add (or) modify the test cases for better testing.

Testing shows presence of defects: We need to test an application with an intension to show defects for which negative testing is the best approach.

Early Testing: Testing should start as early as possible in the SDLC.

Testing is context dependent: We need to select or opt for appropriate testing approach based on the type of application we are testing.

Absence of Errors is a falling: finding and fixing defects absence bug free is impossible.

******* (Fallacy = False statement)**

Static Testing: Verifying if we are developing the right system or not is called static testing. It is also called verification and this static testing can be used to conduct reviews and walk throughs.

Reviews: Examining any project related work or project related work is called reviews.

Types of Reviews:

1) Management Reviews
2) Technical Reviews
3) Code Reviews
4) Test case Reviews (formal , Informal)

Formal Reviews: if any review is conducted with a prior plan and by following proper documentation and procedure then it is are called a formal reviews. Inspections and Audits are the best example of formal reviews.

Informal Reviews: if any review is conducted without following any procedures and documentation then it is known as informal reviews

Walk throughs: knowledge transfer sessions are called walk throughs.

Objective of Reviews: Reviews are helpful to determine:

1) Defects in requirement.
2) Defects in design.
3) Deviations in coding standards.
4) To confirm if the prepared test cases are enough to validate a software or not and also.
5) Reviews help in improving the organization process.

Dynamic Testing: It is a process of checking the source code and it also helps in checking if the application is working as per the expectation or not.. It is also called validation.

*** Levels of Dynamic Testing:** Dynamic testing is carried out at 4 levels.

1) Unit Testing
2) Integration Testing
3) System Testing
4) User Acceptance Testing.

Unit Testing: A smallest portion which can be separated in the source code of the application is called unit (functions, procedures, etc) testing, which is conducted on these units to check the code behind the units which are working as per the expectation or not.

(or)

It is also called module testing and also component testing.

Integration Testing: Once all units are tested the programmers will combine all units and check interactions among the units, which is called integration testing.

Note: Unit testing and integration testing is collectively called White Box Testing.

White box Testing: Testing conducted on the source code by developers to check if the source code is working as per the expectation or not is called white box testing.

*** What is the need of white box testing?**

- As the source code is visible finding the problems and rectifying the problems is so easy for developers.
- The defects that are identified in white box testing are economical to resolve.
- White box testing is helpful to reduce the defects as early as possible.
- It is used to ensure 100% code coverage.

******* White box testing is also called as glass box, structural testing, and clear box testing.**

Black box Testing: Testing is conducted on the application by test engineers or by domain experts to check whether the application is working according to the customer requirements or not. This kind of testing is called black box testing.

What is the need of Black Box Testing?

1) White box testing conducted by developer in a technical perception where as black box testing is conducted by test engineers with an end-user perception.
2) Programmers will conduct white box testing in a positive perception whereas tester will conduct black box testing with a negative perception. This will provide a chance of finding more defects
3) The more defects you identify, you will achieve more results in a quality system.
4) White box testing will not cover non functional areas as non functional requirements. It is very important for live production and is covered under black box testing.
5) The objective of white box testing is 100% coverage whereas the objective of black box testing is 100% customer business requirement coverage.
6) Black Box Testing = System Testing + User Accepting Testing. It is called a Requirement Based Testing (or) Specification Based Testing.

System Testing: Validating the functional and non functional requirements of the system is called system testing.

System testing is broadly classified into two types namely;

1) Functional system testing
2) Non-functional system testing

Functional System testing will be conducted both in a positive perception and also in a negative perception.

Positive Testing (+ve): Testing conducted on the application in a positive approach to determine what system is supposed to do is called a positive testing.

******Note:***
Positive testing helps in checking if the customer requirements are justifying the application or not.

Negative Testing (-ve): Testing a software application with a negative perception to check what system is not supposed to do is called negative testing.

******Note :***
Negative testing is helpful to find defects in the software.

Functional System Testing Approach: is also called Smoke Testing (or) Security Testing. It is a kind of quick test (or) rough test carried out on the application to determine whether the application is testable or not.

Formal Testing: If you tested software application has all the procedures and proper documentation, then it is called a formal testing.

Adhoc Testing: If you test software had not followed any procedures and documentation then it is called adhoc-testing. It is also called informal testing. If you tested the application as per your wish without any pre-defined procedure then it is called an adhoc testing.
Risk Based Testing (or) Priority Based Testing: Identifying the critical functionality in the system and then deciding the order in which the functionalities are to be tested and applying testing procedure and

(or) Conducting testing in the same order is called risk based testing (or) priority based testing.

Re- Testing: Testing functionality repetitively is called re-testing.

Re-testing gets introduced in the following two scenarios.

1) Testing is a functionality with multiple inputs to confirm the business validation are implemented (or) not.

2) Testing functionality in the modified build is to confirm the bug fixers are made correctly (or) not.

Regression Testing: It is a process of identifying various features in the modified build where there is a chance of getting side-effects and retesting these features.

1) The new functionalities added to the existing system (or) modifications made to the existing system .It must be noted that a bug fixer might introduce side-effects and a regression testing is helpful to identify these side effects.

End to End Testing: Testing the overall functionalities of the system including the data integration among all the modules is called end-to-end testing.

Exploratory Testing: Exploring the application and understanding the functionalities adding or modifying the existing test cases for better testing is called exploratory testing.

Monkey Testing: Testing conducted on an application unevenly that is in a zig zag way with an intension of finding tricky defects is called monkey testing.

Non-Functional System Testing: Validating various non functional aspects of the system such as, user interfaces, user friendliness, security, compatibility, load, stress and performance etc. is called non-functional system testing.

Non Functional Testing

UI / GUI Testing:

Validating user interfaces and checking if it is professionally designed or not is called UI Testing.

Check List for UI Testing:

1) It checks if all the basic elements are available in the page or not.
2) It checks the spelling of the objects.
3) It checks alignments of the objects.
4) It checks content display in web pages.
5) It checks if the mandatory fields are highlights or not.
6) It checks consistency in background color and color font type and fond size etc.

Usability Testing: Checks how easily the end users are able to understand and operate the application is called usability testing.

Security Testing: Validates whether all security conditions are properly implemented in the software or not. It is called security testing.

Check List for Security Testing:

1) It checks the secured data such as password credit card cvv number are getting encrypted or not.
2) It checks browser navigators after logout
3) It checks direct URL access for the both secured and non secured pages.
4) It checks for session expired or expiry
5) It checks the view source code option for secured pages.
6) It check for Authorization
7) It checks for Authentication
8) It checks cookies.

Performance Testing: It is a process of measuring various efficiency characteristics of a system such as response time, through put, load, stress transactions per minutes, transaction mix.

Load Testing: Analyzing functional and performance behaviour of the application under various load conditions. It is called load testing.

Stress Testing: Checking the application behavior under stress conditions is called stress testing. In other words reducing the system resources and keeping the load as constant checking and how does the application behave is called stress testing.

Recovery Testing: Checking how the system is able to handle unexpected and unpredictable situations is called recovery testing.

Globalization Testing: Checks if the application has a provision of setting and changing languages date and time format and currency etc. If it is designed for global users, it is called globalization testing.

Localization Testing: Checks default languages currency date and time format etc. If it is designed for a particular locality of users is called Localization testing.

Installation Testing: Checks if new software can be installed successfully or not as per the guidelines given in installation document is called installation testing.

Uninstallation Testing: Checks if we are able to uninstall the software from the system successfully or not, it is called uninstallation testing.

Compatibility Testing: Checks if the application is compatible with a different software and hardware environment or not, it is called compatibility testing.

****

For more software testing tutorials visit – http://softwaretestinghelp.com

Want to learn software testing from the experienced professionals?
Visit – http://softwaretestinghelp.org

Get Premium eBook – Software testing Career package eBook

ISTQB Sure Pass Premium Study Material – ISTQB Complete Study package

Thank you!