SUPER 25 SOFTWARE TESTING

Definitions frequently asked (2 Marks)

1) **Static Testing:** - In static testing code is not executed. Rather it manually checks the code, requirement documents, and design documents to find errors. Main objective of this testing is to improve the quality of software products by finding errors in early stages of the development cycle.
2) **Dynamic Testing:** - The dynamic testing is done by executing program. Main objective of this testing is to confirm that the software product works in conformance with the business requirements
3) **Defect:** - It refers to the several troubles with the software product, with its external behaviour or its internal features. **OR**
   A defect is an error in coding that causes a program to fail or to produce incorrect /unexpected results
4) **Bug:** A bug can be defined as the initiation of error or a problem due to which fault, failure, incident or an anomaly occurs.
5) **Error:** - A human action that produces an incorrect result.
6) **Fault:** - An incorrect step, process, or data definition in a computer program.
7) **Failure:** - A failure is said to occur whenever the external behaviour of a system does not conform to that prescribed in the system specification. A software fault becomes a software failure only when it is activated.
8) **Drivers:** - Drivers are dummy modules that always used to simulate the high-level modules
9) **Stubs:** - Stubs are dummy modules that always used to simulate the low-level modules.
10) **Test Plan: -** A Test Plan is a detailed document that describes the test strategy, objectives, schedule, estimation, deliverables, and resources required to perform testing for a software product. Test Plan helps us determine the effort needed to validate the quality of the application under test. The test plan serves as a blueprint to conduct software testing activities as a defined process, which is minutely monitored and controlled by the test manager.

11) State the classification of defects.

1. Requirement/Specification Defects:

Requirement-related defects arise in a product when one fails to understand what the customer requires.
These defects may be due to the customer gap, where the customer is unable to define his requirements.
Producer gap, where the developing team is not able to make a product as per requirements.

2. **Design Defects**:

- Design defects occur when system components, interactions between system components, interactions between the outside software/hardware, or users are incorrectly designed.
- Design defects generally refer to the way of design creation or its usage while creating a product.

3. **Coding Defects**:

This defect arises when variables are not initialized properly, or variables are not declared correctly, or database is not created properly.
Coding also needs adequate commenting to make it readable and maintainable in future.

4. **Testing Defects**:

These would encompass incorrect, incomplete, missing inappropriate test cases and test procedures.

12. List the levels of testing.

- Following are the levels of testing:
- a) Unit test
- b) Integration test
- c) System test
- d) Acceptance test

Differentiate between frequently asked (4 M)

1) Alpha Testing Beta Testing

Sr.No	Alpha testing.	Beta testing.
1	Performed at developer's site	Performed at end user's site
2	Performed in controlled environment in developers' presence	Performed in uncontrolled environment in developers absence
3	Less probability of finding errors as it is driven by developer	High probability of finding errors as it is used by end user.
4	It is done during implementation phase of software	It is done at the pre-release of the software
5	It is not considered as live application of software	It is considered as a live application of the software.
6	Less time consuming as developer can make necessary changes in given time	More time consuming as user has to report the bugs if any via appropriate channels.
7	Alpha testing involves both white box and black box testing	Beta testing typically uses black box testing only
8	Long execution cycles may be required for alpha testing	Only a few weeks of execution are required for beta testing

2) Verification and validation

S-16. S-18. 4 Marks

Q.1.5.1 Differentiate between verification and validation. (Ref. Sec. 1.5.1)

Parameter	Verification	Validation
Focus	Concentrates on right way to build a system.	Concentrates on output of build system.
Definition	Verification can be defined as it is a process of estimating the intermediary work products of a software development lifecycle to verify that we are in the correct track of creating the end user product.	Validation can be defined as it is the process of evaluating the final product to ensure that the software meets all the specified requirements.
Aim	The aim of Verification is to ensure that the product being develop is as per the requirements and design specifications.	The aim of Validation is to ensure that the product really meets up the user's requirements, and verify whether the specifications be correct in the first place.
Inclusion	Verification contains Reviews, Meetings and Inspections.	Validation contains testing such as black box testing, white box testing, gray box testing etc.
Performer	Verification is conducted by quality assurance team.	Validation is conducted by testing team.
Execution of code	Execution of code is not a part of Verification.	Execution of code is a part of Validation.
Explanation	Verification process gives explanation about whether the outputs are as per the inputs or not.	Validation process gives explanation about acceptance of software by the user or not.
Processes	Plans, Requirement Specifications, Design Specifications, Code, and Test Cases etc. are evaluated in verification.	Testing of actual software is done in validation.

3) Stubs and Drivers

	Stub	Driver
Type	Dummy codes	Dummy codes
Description	Routines that don't actually do anything except declare themselves and the parameters they accept. The rest of the code can then take these parameters and use them as inputs	Routines that don't actually do anything except declare themselves and the parameters they accept. The rest of the code can then take these parameters and use them as inputs
Used in	Top Down Integration	Bottom Up Integration
Purpose	To allow testing of the upper levels of the code, when the lower levels of the code are not yet developed.	To allow testing of the lower levels of the code, when the upper levels of the code are not yet developed.

4) Smoke and Sanity Testing

Sr. No.	Smoke Testing	Sanity Testing
1.	Smoke testing is performed to discover that the critical functionalities of the program are working fine.	Sanity testing is done to check the new functionality / bugs have been fixed.
2.	The objective of this testing is to verify the "stability" of the system in order to proceed with more rigorous testing.	The objective of the testing is to verify the "rationality" of the system in order to proceed with more rigorous testing.
3.	This testing is performed by the developers or testers.	Sanity testing is usually performed by testers.
4.	Smoke testing is usually documented or scripted.	Sanity testing is usually not documented and is unscripted.
5.	Smoke testing is a subset of Regression testing.	Sanity testing is a subset of Acceptance testing.
6.	Smoke testing exercises the entire system from end to end.	Sanity testing exercises only the particular component of the entire system.
7.	Smoke testing is like general health check up.	Sanity Testing is like specialized health check up.

5) Black Box Testing and White Box Testing

Parameter	Black Box Testing	White Box Testing
Applicable levels	It is mainly applicable at higher levels of testing : Acceptance and system testing.	It is mainly applicable at lower levels of testing i.e. unit and integration testing.
Base for testing	Requirement specification is the basis for Black Box Testing.	Detail design is the basis for White Box Testing.
Performed by	It is carried out by software testers.	It is carried out by software developers.
Aim	It aims at what functionality is system performing.	It aims at how system is performing its functionality.
Requirements	Programming knowledge and implementation knowledge is not necessary to carry out BBT.	Programming knowledge and implementation knowledge is necessary to carry out WBT.
Cost	It is less expensive than WBT.	It is more expressive than BBT
Focus on	In black box testing, we do not test the code and focuses on the functionality of software product.	In white box testing, we test the code and focuses on the security of software product and flow of control.
Facility provided	Black box testing provides facility of verifying communication among different modules present in software product.	White box testing does not provide facility of verifying communication among different modules present in software product.

6) Static Testing Tools and Dynamic Testing Tools

– Few points of differences among static and dynamic testing are as follows :

Parameter	Static Testing	Dynamic Testing
Basic	Static Testing is white box testing which is done at early stage if development life cycle. It is more cost effective than dynamic testing.	Dynamic Testing on the other hand is done at the later stage of development lifecycle.
Statement Coverage	Static testing has more statement coverage than dynamic testing in shorter time.	Dynamic Testing has less statement stage because it is covers limited area of code.
Implementation	It is done before code deployment.	It is done after code deployment.
Stage	It is performed in Verification Stage.	It is done in Validation Stage.
Code execution need	This type of testing is done without the execution of code.	This type of execution is done with the execution of code.
Gives	Static testing gives assessment of code as well as documentation.	Dynamic Testing gives bottlenecks of the software system.
Process	In Static Testing techniques a checklist is prepared for testing process.	In Dynamic Testing technique the test cases are executed.
Includes	Static Testing Methods include Walkthroughs, code review.	Dynamic testing involves functional and nonfunctional testing

7) Quality Assurance and Quality Control

Q.1.7.2 Differentiate between Quality Assurance ...

Parameter	Quality Assurance	Quality Control
Definition	QA is a set of activities for ensuring quality in the processes by which products are developed.	QC is a set of activities for ensuring quality in products. The activities focus on identifying defects in the actual products produced.
Focus on	QA aims to prevent defects with a focus on the process used to make the product. It is a proactive quality process.	QC aims to identify (and correct) defects in the finished product. Quality control, therefore, is a reactive process.
Goal	The goal of QA is to improve development and test processes so that defects do not arise when the product is being developed.	The goal of QC is to identify defects after a product is developed and before it's released.
How	Establish a good quality management system and the assessment of its adequacy. Periodic conformance audits of the operations of the system.	Finding and eliminating sources of quality-problems through tools & equipment so that customer's requirements are continually meet.
What	Prevention of quality problems through planned and systematic activities including documentation.	The activities or techniques used to achieve and maintain the product quality, process and service.
Responsibility	Everyone on the team involved in developing the product is responsible for quality assurance.	Quality control is usually the responsibility of a specific team that tests the product for defects.
Example	Verification is an example of QA.	Validation/Software Testing is an example of QC.
Statistical Techniques	Statistical Tools & Techniques can be applied in both QA and QC. When they are applied to processes (process inputs and operational parameters), they are called Statistical Process Control (SPC); and it becomes the part of QA.	When statistical tools & techniques are applied to finished products (process outputs), they are called as Statistical Quality Control (SQC) and comes under QC.

8) Manual Testing and Automated Testing

...Automated testing. (Ref. Sec. 5.1.4) S-18, 4 Marks

Parameter	Manual Testing	Automation Testing
Human interaction	To perform manual testing human interaction is required for test execution.	To perform Automation Testing, human interaction is not required. In automation testing we use tools to run the test cases.
Requirements	To perform manual testing, we need skilled employees, more time and high costs.	To perform automation testing, we require automation tools. Automation testing saves time, cost and manpower. Once test suite is recorded in automation tool then it can easily run test suit.
Application to Test	Any application can be first tested manually. Informal testing like ad-hoc and monkey testing are performed manually.	Automated testing is used to test an application whose requirements are stable at some extent and automation testing mainly used to perform Regression Testing.
Execution of Same test cases	We cannot use manual testing while executing same test suit. Repetitive execution of test cases become boring and error prone.	The most boring part which is of running same test cases repetitively can be performed with the help of automation testing.

Test Case (4/6M)

Railway Reservation System

Test case ID	Test case objective	Input data	Expected result	Actual result	Status
TC1	Login field	Any valid login name (abcxyz)	It should accept the login name	It accepted the login name	Pass
TC2	Password field	Valid password	It should accept the valid password	It accepted the valid password; successful login message	Pass
TC3	Password field	Invalid password	It should not accept the valid password	Message displayed as invalid login or wrong password.	Pass
TC4	Date of journey	Date format not before the current date	It should accept date	Accepted the date	Pass
TC5	Date of return journey	Date format, date greater than the date of journey	It should accept the date	Accepted the date	Pass
TC6	Boarding station	Valid boarding station	It should accept	Accepted the boarding station	Pass
TC7	Train number	Valid train number	It should accept the valid train number	Train number accepted	Pass

1) GUI testing for Amazon and Flipkart

Test case ID	Test case objective	Input data	Expected result	Actual result	Status
TC1	Check cursor position at email or mobile number field	Click on email or mobile number field	Cursor should be placed on the field	Placed the cursor on the field	Pass
TC2	Check cursor position at password field	Click on password field	Cursor should be placed on the password field	Placed the cursor on the password field	Pass
TC3	Check the continue button	Click on continue button	It should redirect to password page	It redirected to the password page.	Pass
TC4	Readability of font	Try to read the contents on login page	Contents should be readable	Contents are readable	Pass
TC5	Testing of spelling of login	Check the spelling of login	Login spelling should be correct	Spelling of Login is correct	Pass
TC6	Testing of hyperlink	Hover the mouse on hyperlink	It should change the cursor and should redirect to respective page on click	Cursor changed and redirects to other page.	Pass

2) Hostel Admission

Test case ID	Test case objective	Input data	Expected result	Actual result	Status
TC1	Student name field	Any valid alphabetical characters (John)	It should accept the name	Students name is accepted	Pass
TC2	Date of birth field	Date format before the current date	It should accept the date less than the current date	It accepted the valid date	Pass
TC3	Gender field	Radio button should be selected. F or M	It should select the proper radio button	Proper radio button is selected	Pass
TC4	Date of admission	Date format not before the current date	It should accept date	Accepted the date	Pass
TC5	Age field	Any numerical data greater than or equal to 16	It should accept the number greater than or equal to 16	Accepted the age	Pass
TC6	Address field	Valid alpha numeric characters	It should accept the address	Accepted the address	Pass
TC7	Pin code	Valid 6 digits numeric format	It should accept the valid pin code	Pin code accepted	Pass

3) Edit Function of Notepad

a. Test plan for windows based Notepad

Test case ID	Test case objective	Input data	Expected result	Actual result	Status
TC1	Test the select all option	Click on select all	All the text should be selected	All the text is selected	Pass
TC2	Cut option	Select the text and click on cut	Selected text should be cut	Selected text is cut	Pass
TC3	Paste option	Click on paste	Contents should be pasted	Contents are pasted	Pass
TC4	Delete option	Select text and click on delete	Contents should be deleted	Contents are deleted	Pass

4) **Prepare and write six test cases for Library Management System of college.**

Test Case No	Test Case Name	Actual Input	Expected output	Actual Output	Status
TC1	User Authentication	1.Enter Username: "22203A0011"	Login should be done successfully	Login is done successfully.	Pass

		2.Enter Password="sandip@1234"			
TC2	Borrowing Book	Enter Book Name: "Data structure using C"	1.Book should be issued to student. 2.Book marked as borrowed	1.Book is issued to student. 2. Book is marked as borrowed in the database.	Pass
TC3	Borrowing Book	Enter Book Name: "123abcd"	It should display message "Enter Valid Book Name"	It is displaying message "Enter Valid book name"	Pass
TC4	Borrowing Book	Enter Book Name: "Data structure and algorithm"	It should display message "Book not available"	It is displaying message: "Book not available"	Pass
TC5	Returning Book	Return a borrowed book	It should mark Book as returned in student and library database.	It is marking Book as returned in student and library database.	Pass
TC6	Fine calculation	Return a book after the due date	It should display calculated fine in student login.	Fine calculated and displayed in student login.	Pass
TC7	Notification after borrowing book	1.Enter book Name: "Data structure and algorithm" 2.Collect book from librarian.	Student should Receive notification for successfully borrowed books	Student receives notification for successfully borrowed books	Pass

5) **Design test cases for Web pages testing of any Web site (take a suitable example).**

Test Case ID	Test case objective	Input data	Expected result	Actual result	Status
TC1	Check cursor position at email or mobile number field	Click on email or mobile number field	Cursor should be placed on the field	Placed the cursor on the field	Pass
TC2	Check cursor position at password field	Click on password field	Cursor should be placed on the password field	Placed the cursor on the password field	Pass
TC3	Check the continue button	Click on continue button	It should redirect to password page	It redirected to the password page.	Pass
TC4	Readability of font	Try to read the contents on login page	Contents should be readable	Content s are readable	Pass
TC5	Testing of spelling of login	Check the spelling of login	Login spelling should be correct	Spelling of Login is correct	Pass
TC6	Testing of hyperlink	Hover the mouse on hyperlink	It should change the cursor and should redirect to respective page on click	Cursor changed and redirects to other page.	Pass

6) **Online mobile recharge**

Test case_id	Test case	Precondition	Test data	Steps to be executed	Expected results	Actual results	Pass/ Fail
1.	Login to application with valid mobile number	User must enter valid mobile no.	Mobile number	1) Enter mobile number 2) Generate OTP on your mobile. Enter OTP	User must successfully login to the web page	Successfully login to the web page	Pass
2.	Circle/state Selection	Mobile number must be registered in state or circle	Select proper state	1) Select proper state	Sate selected	Sate selected	Pass
3.	Enter email-id	User have Email id	Enter email-id	Enter email-id	Accept email id	(note down the results you have observed)	(note down the results you have observed)
4.	Recharge mobile	-	Enter amount	1) Enter amount for recharge	Mobile recharged and SMS received	Mobile recharged and SMS received	Pass

7) **Test cases for online transfer at banking system with respect to client server testing**

banking system. (6 Marks)

Test case_id	Test case	Precondition	Test data	Steps to be executed	Expected results	Actual results	Pass/ Fail
1.	Verify whether the username and password are being accepted or not.	User must registered user.	User id and Password	1) Enter correct user id and password and *CAPTCHA* 2) Generated OTP on your mobile. Enter OTP	Successfully login to the web page	Successfully login to the web page	Pass
2.	Adding Beneficiary	-	Beneficiary Name, Account No., IFSC code, Branch, Add	1) Enter all details correctly	Beneficiary added successfully	Beneficiary added successfully	Pass
3.	Validating Beneficiary	-	OTP received on registered mobile	Enter correct OTP	Validated Beneficiary	Validated Beneficiary	Pass
4.	Online money Transfer	-	Enter amount and Transaction password	1) Enter amount and Transaction correct password	Money Transferred successfully with transaction id	Money Transferred successfully with transaction id	Pass

8) **For withdraw an amount from ATM and prepare a defect report**

Ans. :

Test case_id	Test case	Precondition	Test data	Steps to be executed	Expected results	Actual results	Pass/ Fail
1.	Verify that when card in inserted in ATM, pin should be asked from user.	-	PIN an Debit card	1. Insert debit card 2. Enter PIN	Card accepted ask for enter Pin and successfully accepted	Card accepted ask for enter Pin and successfully accepted	Pass

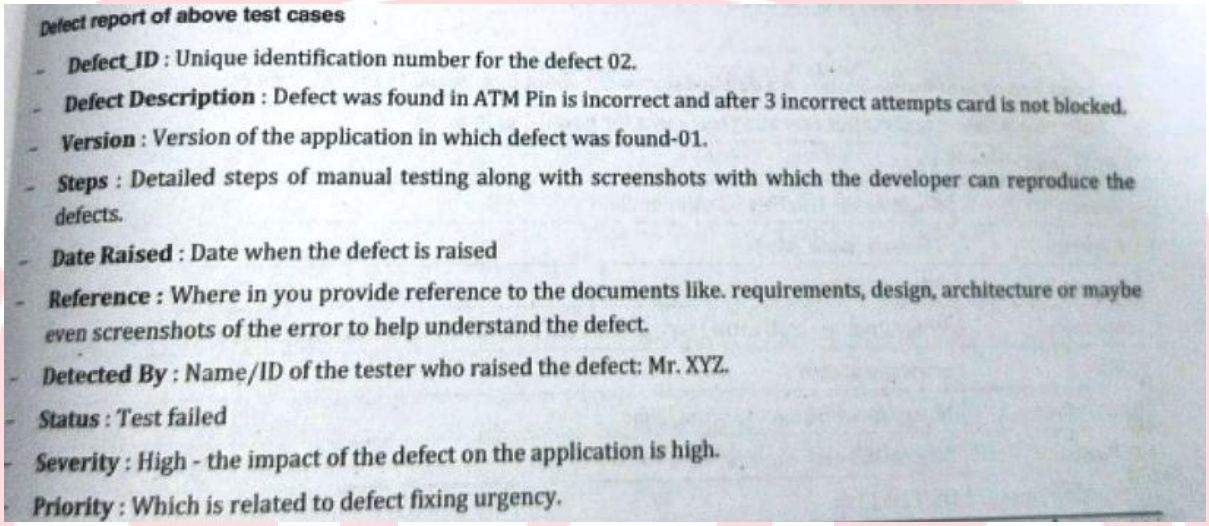
Test case_id	Test case	Precondition	Test data	Steps to be executed	Expected results	Actual results	Pass/ Fail
2.	Verify that when user enters incorrect pin for a particular number of times, the card is blocked.	-	Incorrect pin	1. Insert debit card 2. Enter PIN	Pin is incorrect and after 3 incorrect attempts card blocked	Pin is incorrect and after 3 incorrect attempts card is not blocked	Fail
3.	Verify that when user enters correct pin, the user details should be displayed on ATM screen.	-		Enter correct pin	PIN accepted and the user details should be displayed on ATM screen.	PIN accepted and the user details should be displayed on ATM screen.	Pass
4.	Verify that ATM machine asks to user for the amount to be withdrawn.	-	withdraw amount	Enter amount	Amount withdrawn	Amount withdrawn	Pass
5.	Verify that if use enters amount greater than daily withdraw limit, error message is displayed.	-	Amount	Enter amount.	Amount greater than daily withdraw limit, error message is displayed.	Amount greater than daily withdraw limit, error message is displayed.	Pass
	Verify that if doesn't enter amount in round off digits, error message is displayed.	-	Amount	Enter amount.	Verify that if doesn't enter amount in round off digits, error message is displayed.	Verify that if doesn't enter amount in round off digits, error message is displayed.	Pass

9) **Design test cases for MS Word application using an Automation tool.**

Test Case ID	Test case objective	Input data	Expected result	Actual result	Status
TC1	Check whether Undo in Edit main menu undoes the previous action		Previous action should be undone	Previous action was undone	Pass
TC2	Checks whether the Undo button in right click context menu undoes the previous action		Previous action should be undone	Previous action was undone	Pass
TC3	Checks whether Undo button in the Edit main menu is disabled when there is not any previous actions		Undo Button should be disabled	Undo Button was disabled	Pass
TC4	Checks whether Undo button in right context menu is disabled when there are not any previous actions		Undo Button should be disabled	Undo Button remained disabled	Pass
TC5	Checks whether hotkey (CTRL+Z) response when there is no any of previous actions		No response is expected	No response	Pass
TC6	Checks whether the Cut options in Edit main menu cuts the selected text		Selected text should be cut	Selected text was cut	Pass
TC7	Checks whether the Cut options in Edit Menu is disabled when no texts are selected		Cut Options should be disabled	Cut Option Was Disabled	Pass

Defect Reports

1) **Prepare defect report after executing test cases for withdrawn of amount from ATM machine.**



Defect report of above test cases

- **Defect_ID** : Unique identification number for the defect 02.
- **Defect Description** : Defect was found in ATM Pin is incorrect and after 3 incorrect attempts card is not blocked.
- **Version** : Version of the application in which defect was found-01.
- **Steps** : Detailed steps of manual testing along with screenshots with which the developer can reproduce the defects.
- **Date Raised** : Date when the defect is raised
- **Reference** : Where in you provide reference to the documents like. requirements, design, architecture or maybe even screenshots of the error to help understand the defect.
- **Detected By** : Name/ID of the tester who raised the defect: Mr. XYZ.
- **Status** : Test failed
- **Severity** : High - the impact of the defect on the application is high.
- **Priority** : Which is related to defect fixing urgency.

2) **Prepare defect report for login field of email application**

- Defect Report in Software Testing is a detailed document about bugs found in the software application Following is Defect report after executing test cases for **Email-log in form.**
- ID number- #123
- Name- login form - Unable to login Email
- Reporter- Person's name (xyz)
- Submit Date- 03/01/2023
- Summary- When I put my mail id and password, I am unable to login while login credentials are right.
- URL- www.gmail.com
- Screenshot- https://accounts.google.com/signin/
- Platform- AngularJS
- Operating System- OS X 10.12.0
- Browser- Chrome 53
- Severity- Major
- Assigned to- /
- Priority- High
    - **Description** When I put mail id and password, I am unable to login while login credentials are right.
    - **Steps to reproduce** > go to the www.gmail.com > Click on login button > Put Right mail id and password and click next. > and take Screenshot.
    - **Expected result** The mail account should logged in after putting the right mail id and password.
    - **Actual result** The mail account is not logging in after putting the right details.

Test Plans

1) **Prepare test plan for "Cam Scanner' which is installed on mobile.**
2) **Test plan along with test cases for creating a saving account at bank**
3) Prepare test plan for Notepad application
4) Prepare a test plan along with the test cases for the MS Excel option “Save AS”.
5) **Design a test plan along with the test cases for edit function in notepad**

Theory questions(4/6 M)

1) **Define Boundary value analysis with suitable example**

- **Boundary Value Analysis**
- Its a black box testing method
- It Checks bugs at boundary of input
- It Checks both valid and invalid boundary partitions
- BVA is negative or stress testing

- Most of the defects in software products hover around conditions and boundaries.
- Boundary value analysis is another black box test design technique, and it is used to find the errors at boundaries of input domain rather than finding those errors in the centre of input.
- Each boundary has a valid boundary value and an invalid boundary value.
- Test cases are designed based on both valid and invalid boundary values. Typically, we choose one test case from each boundary.

The basic idea in boundary value testing is to select input variable values at their:

1. Minimum
2. Just below the minimum
3. Just above the minimum
4. Just below the maximum
5. Maximum
6. Just above the maximum

Example: Input Box should accept the Number 1 to 10

- Here we will see the Boundary Value Test Cases

Test Scenario Description	Expected Outcome
Boundary Value = 0	System should NOT accept
Boundary Value = 1	System should accept
Boundary Value = 2	System should accept
Boundary Value = 9	System should accept
Boundary Value = 10	System should accept
Boundary Value = 11	System should NOT accept

-

2) **Draw a diagram for defect life cycle and write example for defect template.**

- **Defect Life Cycle:**

a. Defect life cycle, also known as **Bug Life cycle**.

b. It is the **journey of a defect cycle**, which a defect goes through during its lifetime.

c. It **varies** from organization to organization and project to project.

d. It is governed by the software testing process.

e. It depends upon the tools used.

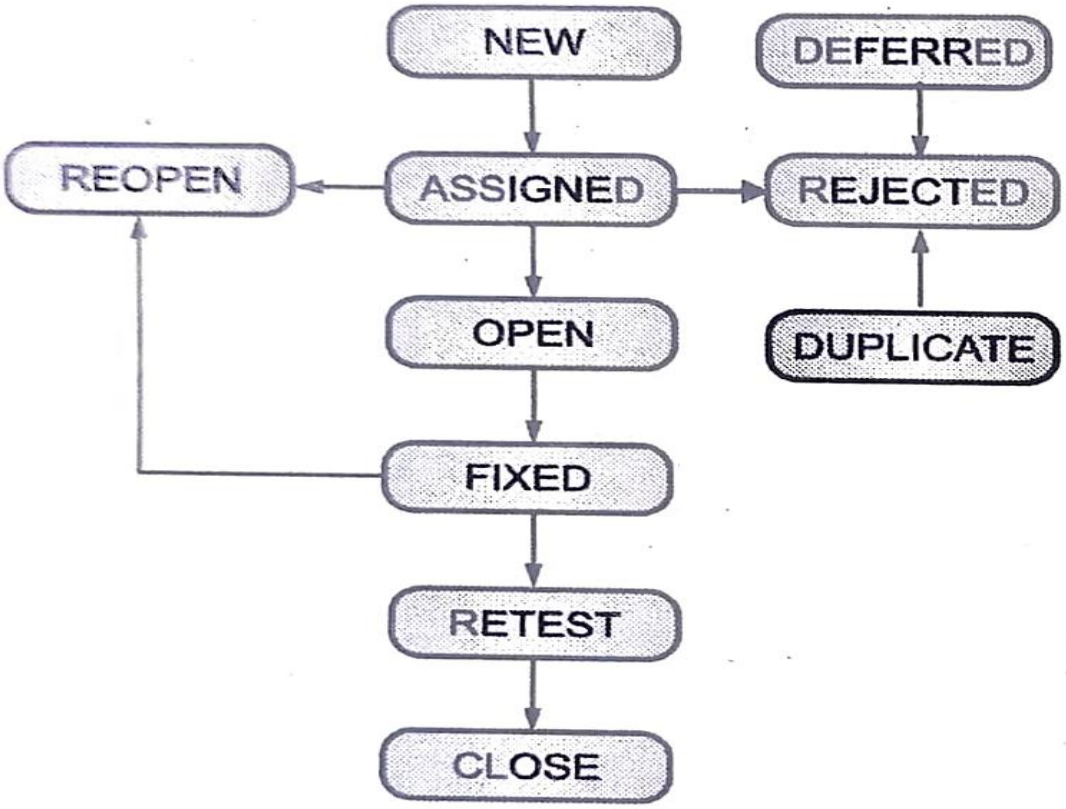


Fig. 4.3.1 : Defect Life Cycle

- Defect Life Cycle includes following stages:
  a. **New**: When a defect is logged and posted for the first time.
  b. **Assigned**: Once the bug is posted by the tester, the lead of the tester approves the bug and assigns the bug to developer team. Defect can be assigned to developer who owns the functionality or can be assigned to development lead who will further move the defect to developer.

c. **Open**: It is the state when developer starts analysing and working on the defect fix.

d. **Fixed**: When developer makes necessary code changes and verifies the changes then he/she can make bug status as “Fixed”.

e. **Retest**: At this stage the tester do the retesting of the changed code which developer has given to him to check whether the defect got fixed or not.

f. **Reopened**: If the bug still exists even after the bug is fixed by the developer, the tester changes the status to "reopened". The bug goes through the life cycle once again.

g. **Deferred**: The bug, changed to deferred state means the bug is expected to be fixed in next releases. the bug may not have major effect on the software.

h. **Rejected**: If the developer feels that the bug is not genuine, developer rejects the bug. Then the state of the bug is changed to “rejected".

i. **Duplicate**: If the bug is repeated twice or the two bugs mention the same concept of the bug, then the recent/latest bug status is changed to "duplicate “.

j. **Closed**: Once the bug is fixed, it is tested by the tester. If the tester feels that the bug no longer exists in the software, tester changes the status of the bug to "closed”. This state means that the bug is fixed, tested and approved.

3) **State the need of automated testing tools.**

- An automated testing tool can playback pre-recorded and predefined actions, compare the results to the expected behaviour and report the success or failure of these to a test engineer.
- Once automated tests are created, they can easily be repeated, and they can be extended to perform tasks impossible with manual testing.
- Automated Software Testing Saves Time and Money.
- Software tests must be repeated often during development cycles to ensure quality.
- Every time source code is modified software tests should be repeated.

- For each release of the software, it may be tested on all supported operating systems and hardware configurations. Manually repeating these tests is costly and time consuming
- Once created, automated tests can be run repeatedly at no additional cost, and they are much faster than manual tests.
- Testing Improves Accuracy, Even the most conscientious tester will make mistakes during monotonous manual testing.
- Automated tests perform the same steps precisely every time they are executed and never forget to record detailed results. Describe any four limitations of manual testing. They can even be run on multiple computers with different configurations.
- Automated software testing can look inside an application and see memory contents, data tables, file contents, and internal program states to determine if the product is behaving as expected.

4) **Give the objectives of software testing.**

● To find any defects or bugs that may have been created when the software was being developed

● To increase confidence in the quality of the software

● To prevent defects in the final product

● To ensure that end product meets customer requirements as well as specifications

● To provide customers with a quality product and increase their confidence in the team.

5) **State the Entry and Exit criteria for the software testing.**

- **Entry criteria**
- Entry criteria are the condition or the set of conditions, which should exist or be met in order to start a process.
- Some of the conditions or situations, which may be seen as an entry criterion for the initiation of testing activities.
- • Requirements should be clearly defined and approved.
- • Test Design and documentation plan is ready.
- • Availability of the test environment supporting necessary hardware, software, network configuration, settings, and tools for the purpose of test execution.
- • Testers are trained, and necessary resources are available.
- • Availability of proper and adequate test data (like test cases).
- • It depends upon which software development model is used.
- **Exit criteria** Exit Criteria is often viewed as a single document concluding the end of a life cycle phase.
- Some of the conditions or situations which may be seen as an exit criterion for testing activities.
- • Testing Deadline
- • Completion of test case execution.
- • Completion of Functional and code coverage to a certain point.

- • Bug rates fall below a certain level and no high priority bugs are identified.
- • Management decision.

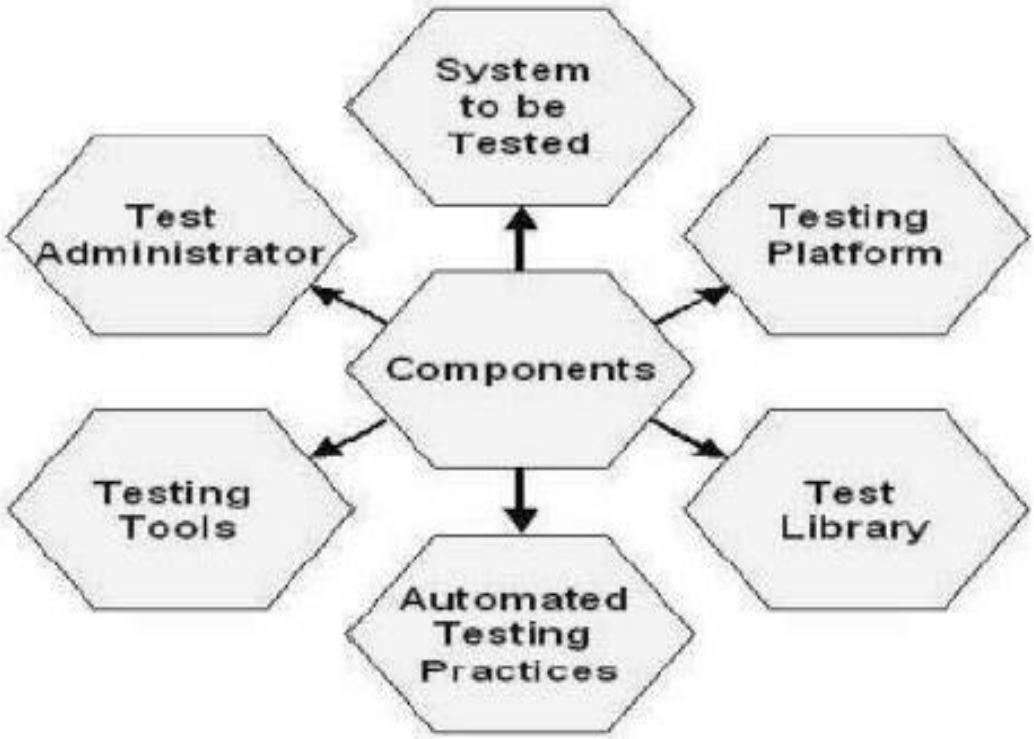
6) **Describe the "Test Infrastructure' components with diagram.**

- The testing infrastructure consists of the testing activities, events, tasks, and processes that immediately support automated, as well as manual, software testing.
- The stronger the infrastructure the more it provides the stability, continuity and reliability of the automated testing process.
- The testing infrastructure includes:
- Test plan.
- Test cases.
- Baseline test data.
- A process to refresh or roll back to baseline.
- A dedicated test environment.
- A dedicated test lab.
- Integration groups and process.
- Test case database, to track and update both automated and manual testing.
- A way to prioritize, or rank, test cases per test cycle.
- Coverage analysis.
- Defect tracking database.
- Risk management metrics/process.
- Version control system.
- Configuration management process.
- Metrics to measure improvement.
- A test case database (TCDB) (additional)
- A test case database captures all the **relevant information** about the test cases in an organization. Some of the entities and the attributes are given in following table

Entity	Purpose	Attributes
Test case	Records all the —staticl information about the tests	• Test case ID • Test case name (filename) • Test case owner • Associated files for the test case
Test case- product cross reference	Provides a mapping between the tests and the corresponding product features ; enables identification of tests for a given feature	• Test case ID • Modulate ID
Test case run history	Gives the history of when a test was run and what was the result; provides inputs on selection of tests for regression runs (see chapter 8)	• Test case ID • Run date • Time taken • Run status (success/failure)
Test case – Defect cross reference	Gives details of test cases introduced to test certain specific defects detected in the product ;provides inputs on the selection of tests for regression runs	• Test case ID • Defect reference# (points to a record in the defect repository)

- Defect Repository

Entity	Purpose	Attributes
Defect details	Records all the — static information about the tests	• Defect ID • Defect priority /severity • Defect description • Affected product(s) • Any relevant version information (for example, OS version) • Customers who encountered the problem (could be reported by the internal testing team also) • Date and time of defect occurrence
Test case- product cross reference	Provides a mapping between the tests and the corresponding product features ; enables identification of tests for a given feature	• Test case ID • Modulate ID
Test case run history	Gives the history of when a test was run and what was the result; provides inputs on selection of tests for regression runs (see chapter 8)	• Test case ID • Run date • Time taken • Run status (success/failure)
Test case – Defect cross reference	Gives details of test cases introduced to test certain specific defects detected in the product ;provides inputs on the selection of tests for regression runs	• Test case ID • Defect reference# (points to a record in the defect repository)



Test Infrastructure

-
-

7) **State the contents of "Test Summary Reports" used in test reporting.**

- **Preparing test summery:**
- Test needs continuous communication between test team and other teams
- Test Reporting: Test reporting is a means of achieving communication through the testing cycle.
- There are **3 types** of test reporting.
- **1. Test incident report:**

2. Test cycle report:

3. Test summary report:

- A test summary report has the role of a comprehensive documentation of the testing activities conducted throughout the software development life cycle (SDLC).
- The final step in a test cycle is to recommend the suitability of a product for release.
- A report that summarizes the result of a test cycle is the test summary report.
- There are two types of test summary report:

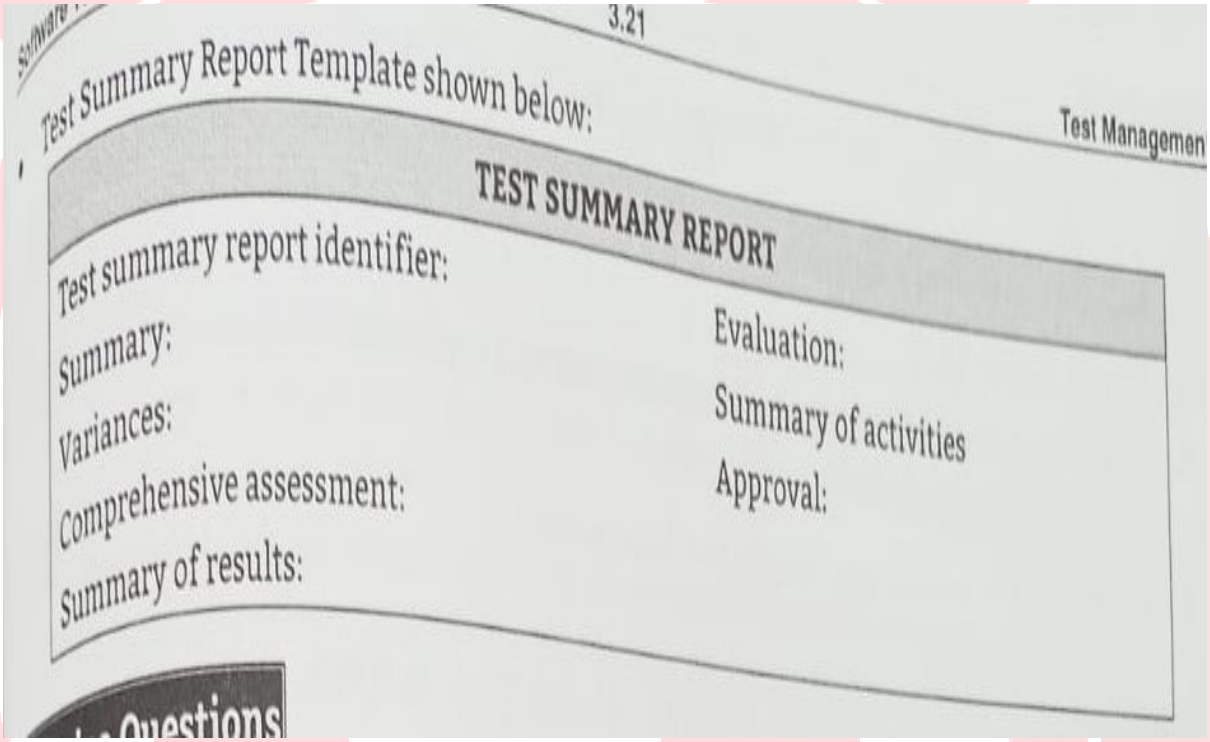
1. **Phase wise test summary**, which is produced at the end of every phase

2. **Final test summary report**.

- - **Aspects covered by Test Summary Report:**
- **Project Information**: name, version, date, time

- **Test Objective**: purpose of testing(every testing has different purpose).
- **Test Summary**: summary of testing process(executed, passed, failed, blocked, comments from testers).
- **Defects**: bugs and their status-open, closed, resolved-their severity and priority.
- **Format Of Test Summary Report**
- The standard for reporting is IEEE-829:1998

- # Test summery template

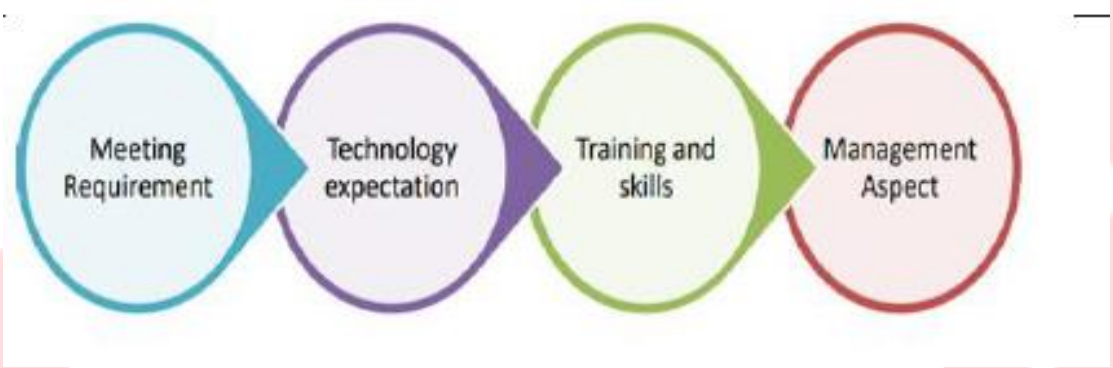


- 1. Test Summary Report Identifier
- 2 Description: Identify the test items being reported in this report with test id 3 Variances: Mention any deviation from test plans, test procedures, if any. 4 Summary of results: All the results are mentioned here with the resolved incidents and
- their solutions. 5 Comprehensive assessment and recommendation for release should include: Fit for release assessment and recommendation of release.

8) **Enlist the factors considered for selecting a testing tool for test automation**

- It's a challenging job for test managers and test architects.
- **Features for selecting static test tools:**
- **Flexibility and Ease of Use:**
- Tool should be easy to use and its training and user adaptation time should be very less.
- Testing tool should be configurable enough to support model variance of project.
- **Support for End-to-End Traceability**
- It is important for testers to trace back all their work.
- Traceability increases efficiency of measuring quality of a project.
- It allows organizations to track coverage of both

- 1st requirements and
- 2nd test cases.
- **Real Time reports and Dashboards**
- Projects fail due to lack of proper visualization of analytical data related to projects progress.
- If no tool is available the whole process is dependent on manual interactions making it error prone.
- So, tool should have real time support and dashboard for keeping stakeholders updated with latest status of progress and access quality at every step.
- **Support for Test Automation**
- Due to fierce market conditions, In today's world, automation has become mandatory for organizations.
- A testing tool must have the support for managing test automation scripts from a single repository.
- Features like Central execution of test scripts, automatic capturing of test results and making it visible from central platform are necessary
- **Integration with other phases of application lifecycle**
- Testing is integral part of entire project lifecycle.
- Testing should be involved in each phase of lifecycle.
- So centralized status update on project's progress and quality can be achieved.
- Seamless integration between testing and lifecycle tools achieves continuous integration and continuous delivery
- **OR**



Selection criteria for testing tool

1. Meeting requirements

• There are plenty of tools available in the market, but rarely do they meet all the requirements of a given product or a given organization.

• Evaluating different tools for different requirements involve significant effort, money, and time.

• The tool must match its intended use.

• Wrong selection of a tool can lead to problems like lower efficiency and effectiveness of testing may be lost. Selection criteria for testing tool

2. Technology expectations:

• Test tools in general may not allow test developers to extends/modify the functionality of the framework

• So, extending the functionality requires going back to the tool vendor and involves additional cost and effort.

• Different phases of a life cycle have different quality-factor requirements. Tools required at each stage may differ significantly. Selection criteria for testing tool

3. Training/skills:

• While test tools require plenty of training, very few vendors provide the training to the required level.

• Organization level training is needed to deploy the test tools.

- • As the user of the test suite are not only the test team but also the development team and other areas like configuration management.

• If the testers do not have proper training and skill, then they may not be able to work effectively. Selection criteria for testing tool

4. Management aspects:

• A test tool increases the system requirement and requires the hardware and software to be upgraded.

• This increases the cost of the already- expensive test tool.

• Select affordable tools. Cost and benefits of various tools must be compared before making final decision

9) Describe graphical user interface (GUI) testing and its important traits

- There are two types of interfaces for a computer application.
- **Command Line Interface** is where you type text and computer responds to that command.
- GUI stands for **Graphical User Interface** where you interact with the computer using images rather than text.
- GUI testing is the process of testing the system's Graphical User Interface of the Application Under Test. GUI testing involves checking the screens with the controls like menus, buttons, icons, and all types of bars - toolbar, menu bar, dialog boxes and windows, etc.
- GUI is what user sees. A user does not see the source code. The interface is visible to the user. Especially the focus is on the design structure, images that they are working properly or not.

- GUI Testing Guidelines

- 1. Check Screen Validations.
- 2. Verify All Navigations .
- 3. Check usability Conditions.
- 4. Verify Data Integrity.
- 5. Verify the object states .
- 6. Verify the date Field and Numeric Field Formats .
- 7. Ensure messages displayed accurately
- 8. Validate font size
- 9.Check alignment
- 10. Check quality and clarity of image
- 11. Check GUI for different screen resolutions

- Advantages of GUI Testing:

- Good GUI improves feel and look of the application; it psychologically accepts the application by the user.
- GUI represents a presentation layer of an application. Good GUI helps an application due to better experience of the users.

- Consistency of the screen layouts and designs improves usability of an application.
- Disadvantages of GUI Testing:
- When number of pages is large and number of controls in a single page is huge requires more memory resources.
- Special application testing like those made for blind people or kids below age of five may need special training for testers.
- Limited or no access to source code –makes process of testing difficult

10) **Describe test deliverables in details.**

- Test Deliverables are the artifacts which are given to the stakeholders of software project during the software development lifecycle.
- There are different test deliverables at every phase of the software development lifecycle.
- Some test deliverables are provided before testing phase, some are provided during the testing phase and some after the testing cycles is over.
- The deliverables include following:
- **The Test Plan itself** (Master test plan, Phase Test Plan, etc.)
- **Test case design specifications**.
- **Test cases** as well as **any automation** which is mentioned in the plan.
- **Test logs** generated by executing the tests.
- **Test summery reports**.
- The **test plan** describes the overall **method to be used to verify** that the software meets the product specification and the customer's needs.
- **It includes** the quality objectives, resource needs, schedules, assignments, methods etc.
- **Test cases** list the specific **items that will be tested** and describe the **detailed steps** that will be followed to verify the software.
- **Bug reports** describe the problems found as the test cases are followed.
- **Test tools and automation** are listed and described which are used to test the software.
- If the team is using **automated methods** to test software, the tools used, **either purchased or written in-house, must be documented**.
- **Metrics, statistics, and summaries** convey the progress being made as the test work progresses. They take the form of **graphs, charts, and written reports.**
- • **Milestones:**
- * milestones are the **dates of completion** given for various tasks to be performed in testing.
- * These are thoroughly **tracked by the test manager** and are kept in the documents such as **Gantt charts, etc.**

11) **Describe load testing and stress testing with suitable example.**

- **Load testing**
- This testing specifies **performance of system** under real-life load conditions.

- It measures **websites quality of service performance** based on actual customer behaviour.
- Load Testing is type of performance testing to check system with **constantly increasing the load** on the system until the time load is reaches to its **threshold value**.
- Here **Increasing load means** increasing number of concurrent users, transactions & check the **behaviour** of application under test.
- It is normally carried out underneath **controlled environment** in order to distinguish between two different systems.
- The main purpose of load testing is to monitor the **response time** and staying power of application when system is performing well under heavy load.
- The successfully executed load testing is only if the specified test cases are executed without any error in allocated time.
- Load testing is testing the **software under customer expected load.**
- In order to perform load testing on the software you **feed it all that it can handle**. Operate the software with **largest possible data files.**
- If the software operates on **peripherals** such as printer, or communication ports, **connect as many as you can**.
- If you are testing an internet server that can handle **thousands of simultaneous connections**, do it. With most software it is important for it to **run over long periods**.
- Some software's should be able **to run forever without being restarted**. So Time acts as a important variable.
- Load testing can be best applied with the help of **automation tools**.
- **Simple examples of load testing**:
- Testing **printer** by sending **large job**.
- Editing a **very large document** for testing of **word processor.**
- **Continuously reading and writing** data into hard disk.
- Running **multiple applications** simultaneously on server.
- Testing of mail server by **accessing thousands of mailboxes.**
- In case of **zero-volume** testing & system fed with **zero load**.
- **Advantages of Load Testing:**
- Discovery of **bottleneck** before deployment.
- Enhance the **scalability** of a system.
- Reduced **risk for system down** time.
- Improved **customer satisfaction**.
- **Reduced failure cost**.

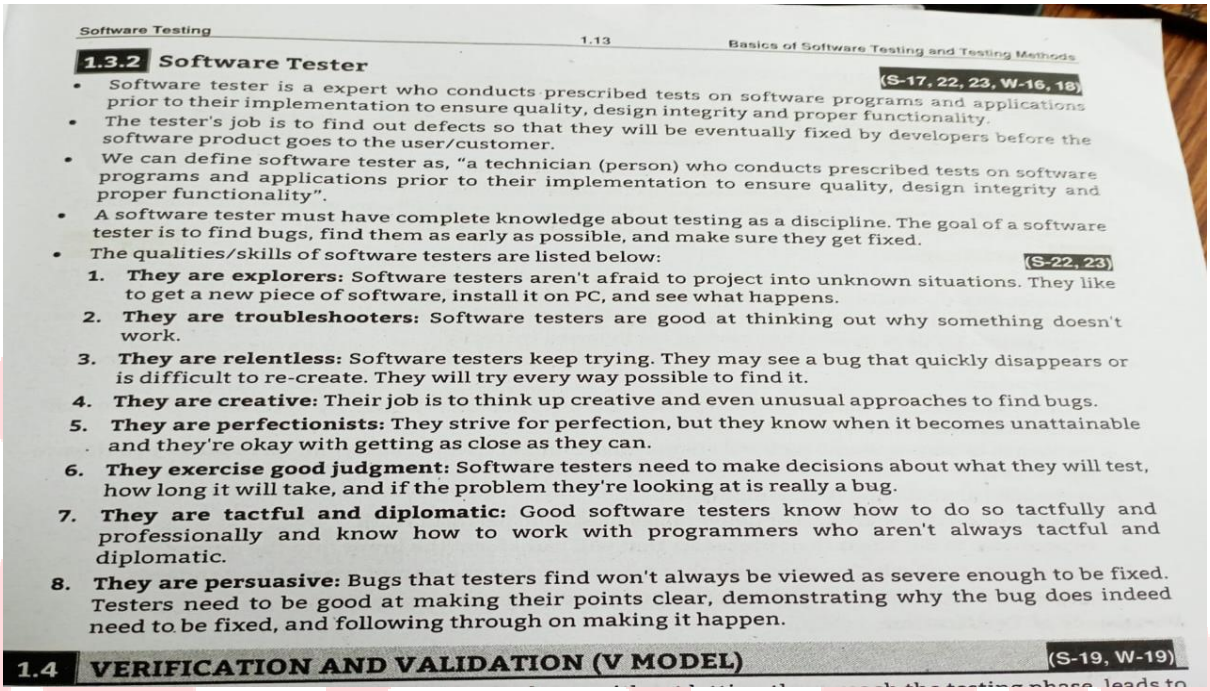
- Stress testing:

- Stress Testing is performance testing type to check the stability of software when hardware resources are not sufficient like CPU, memory, disk space etc.
- It is performed to find the **upper limit capacity** of the system and also to determine how the system performs if the current load goes well **above the expected maximum**.
- Main parameters to focus during Stress testing are **"Response Time" and "Throughput".**
- Stress testing is **Negative testing** where we load the software with large number of concurrent users/processes which cannot be handled by the systems hardware resources.
- This testing is also known as **Fatigue testing**
- Stress testing is **testing the software under less-than-ideal conditions**.

- So subject your software to **low memory, low disk space, slow cps, and slow modems and so on**. Look at your software and determine what external resources and dependencies it has.
- Stress testing is simply **limiting** them to **bare minimum**. With stress testing **you starve the software**.
- For e.g. Word processor software running on your computer with all available memory and disk space, it works fine. But if the system runs **low on resources** you had a greater potential to expect a bug. Setting the values to zero or near zero will make the software execute different path as it attempts to handle the **tight constraint**. Ideally the software would run without crashing or losing data.
- **Advantages of Stress testing**
- It helps developer to know system functionalities and capabilities to handle **multiple tasks at a time**.
- Helps to control and manage bugs
- Allows establishing application- monitoring triggers to warn about incoming failures
- Determines **side effects** of common h/w or supporting application failure.

12) Describe any four skills of software tester.

- **Skills Required for Tester**
- • Communication skills
- • Domain knowledge
- • Desire to learn
- • Technical skills
- • Analytical skills
- • Planning
- • Integrity
- • Curiosity
- • Think from users perspective
- • Be a good judge of your product



Software Testing 1.13 Basics of Software Testing and Testing Methods

1.3.2 Software Tester (S-17, 22, 23, W-16, 18)

- Software tester is a expert who conducts prescribed tests on software programs and applications prior to their implementation to ensure quality, design integrity and proper functionality.
- The tester's job is to find out defects so that they will be eventually fixed by developers before the software product goes to the user/customer.
- We can define software tester as, "a technician (person) who conducts prescribed tests on software programs and applications prior to their implementation to ensure quality, design integrity and proper functionality".
- A software tester must have complete knowledge about testing as a discipline. The goal of a software tester is to find bugs, find them as early as possible, and make sure they get fixed.
- The qualities/skills of software testers are listed below: (S-22, 23)

1. **They are explorers:** Software testers aren't afraid to project into unknown situations. They like to get a new piece of software, install it on PC, and see what happens.
2. **They are troubleshooters:** Software testers are good at thinking out why something doesn't work.
3. **They are relentless:** Software testers keep trying. They may see a bug that quickly disappears or is difficult to re-create. They will try every way possible to find it.
4. **They are creative:** Their job is to think up creative and even unusual approaches to find bugs.
5. **They are perfectionists:** They strive for perfection, but they know when it becomes unattainable and they're okay with getting as close as they can.
6. **They exercise good judgment:** Software testers need to make decisions about what they will test, how long it will take, and if the problem they're looking at is really a bug.
7. **They are tactful and diplomatic:** Good software testers know how to do so tactfully and professionally and know how to work with programmers who aren't always tactful and diplomatic.
8. **They are persuasive:** Bugs that testers find won't always be viewed as severe enough to be fixed. Testers need to be good at making their points clear, demonstrating why the bug does indeed need to be fixed, and following through on making it happen.

1.4 VERIFICATION AND VALIDATION (V MODEL) (S-19, W-19)

-

13) Describe defect management process with neat diagram

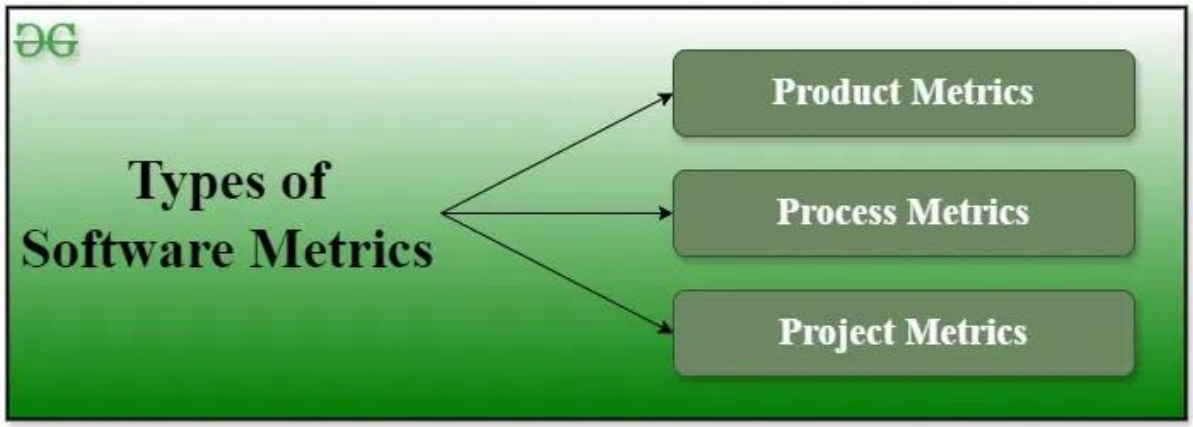
- ## Defect Management Process:
- **Defect Prevention**: it is efficient and effective in reducing the number of defects.
- It is also cost effective to fix defects found during early stages.
- Most organizations conduct **Defect Discovery, Defect Removal and then Process Improvement** which is collectively known as **Defect Management Process.**
- ## Goals of defect management
- Prevent The Defect
- Early Detection
- Minimize The Impact
- Resolution of The Defect
- Process Improvement

-
- ## Defect Prevention
- Best mechanism to remove defects in early stages

- This is cost effective.
- But it is difficult to eliminate all defects at early stages but their impact can be minimized.
- **Steps** involved in defect prevention:
- **Identify Critical Risk**: risks having big impact on the system.
- **Estimate Expected Impact**: calculate amount of financial impact.
- **Minimize Expected Impact**
- : try to minimize or eliminate the risks.
- **Deliverable Baseline**
- When deliverables (system, product or document) reaches its predefined milestones then we can say deliverable is a baseline
- If deliverable moves from one stage to another then **defect** in the system also carried forward to next milestone or stage.
- As issues are identified after completion of earlier milestones they are **called as defects**.
- Finally deliverables are baselined when **changes** into deliverables are **done** and possible **defects** are **identified and fixed**.
- **Defects** which are found before achieving the milestones are **not called as defects** so despite of having defects, code is called as baselined and ready for next milestone
- **Defect Discovery**
- It is almost **impossible to remove all defects** but defects can be identifies at earlier stages before they become costlier.
- **Defects are discovered** means defects are accepted by development team.
- **Steps** involved in Defect Discovery:
- **Find a defect**: Identify defect before it becomes major problem to the system
- **Report Defect**: Testing team makes development team aware about the issue
- **Acknowledge Defect**: Development team acknowledges the issue and fixes it if it is a valid defect
- **Defect Resolution**
- Development team has to proceed for resolution of the defect.
- **Steps** involved in defect resolution:
- **Prioritize the risk**: Defects having more impact on the system are fixed on highest priority.
- **Fix the defect**: higher priority defects are fixed first and lower priority defects are fixed at the end.
- **Report the resolution**: it is the development team's responsibility to ensure that testing team is aware when the defects are going for a fix and how the defect has been fixed.
- **Process Improvement**
- Though the defects are resolved on the basis of priority it does not mean that lower priority defects are not important and not much impacting the system.
- From process point of view all defects are identified are same as a critical defect.
- For process improvement everyone in the project needs to look back and check from where the defect was originated.
- Based on that, changes are made in validation process, baselining document, review process.
- Which will help to catch defect early and will be less expensive

14) Define metrics and measurements. Describe three types of metrics

- A Metric is a measurement of the degree that any attribute belongs
- to a system, product or process.
- For example the number of errors per person hours would be a
- metric. Thus, software measurement gives rise to software metrics.
- A measurement is an indication of the size, quantity, amount or
- dimension of a particular attribute of a product or process. For
- example the number of errors in a system is a measurement.
- A Metric is a quantitative measure of the degree to which a system,
- system component, or process possesses a given attribute.
- Metrics can be defined as “STANDARDS OF MEASUREMENT”.
- Software Metrics are used to measure the quality of the project.
- Simply, Metric is a unit used for describing an attribute. Metric is a
- scale for measurement



-
- **Types of Metrics**
- **Process Metrics:**
- It is used to
- improve the
- efficiency of the process in the SDLC (Software Development Life Cycle).
  - These metrics are, in general, developed for the purpose of monitoring the progress of testing, status of design and development of test cases, and outcome of test cases after execution.
  -
- **Product Metrics:** It is used to tackle the quality of the software product.
  - Product metrics gives information regarding the testing status of a software product.
  - Data required for such metrics are also generated in the process of testing and may help us to know the quality of a product.
  -
- **Project Metrics:** It measures the efficiency of the team working on the project along with the testing tools used.
  -
-

15) What is test plan ? What is its need ? List test planning activities.

- Test plan is a document describing the **scope, approach, recourses and schedule** of intended test activities.
- It is the basis for formally testing any s/w or product

- It identifies amongst other test items:
- the features to be tested,
- the testing tasks,
- who will do each task,
- degree of tester independence,
- the test environment,
- the test design technique,
- entry exit criteria,
- any risk requiring contingency(future event) planning to overcome risk,
- Test plan is a record of all testing process,
- **Master test plan** – addresses **multiple test levels**
- **Phase Test Plan** – Typically address **one test phase.**
- **Following are the activities done as a part of test planning:**
- Preparing a test plan
- Deciding a test approach
- Setting up criteria for testing
- Identifying the responsibilities
- Staffing and testing need
- Resource requirement
- Test deliverables
- Testing tasks
- **Preparing a Test Plan/How to prepare test plan**
- Test plan is to be **started early in STLC**.
- It has to be **short , easy to understand and up-to-date.**
- **Test lead** prepares test plan(testers are also involved).
- Once the plan is ready tester will **write test cases**.
- **Sections of test plan template**
- Selection of test plan document as per IEEE829 standard
- **Test plan identifier**: Test ID
- **References**: List of supporting documents
- **Introduction**: The purpose and scope of project
- **Test Items**: List of test items to be tested
- **Features to be tested**: (login page, dash board etc)
- **Features not to be tested**: features which will be removed later
- **Deciding a test approach**: strategy of testing
- **Setting up criteria for Testing**: (Pass/Fail)
- **Suspension Criteria**: When to stop testing
- **Test Deliverables**: Documents to be delivered at each phase
- **Testing Tasks**: List of tasks need to be completed
- **Resource Requirements**: h/w, s/w, tools etc
- **Identifying Responsibilities**: roles and responsibilities of each test tasks
- **Staffing and Training Needs**: to improve skills
- **Schedule**: complete details from start to finish
- **Risks and Contingencies**: probability of risk and contingency to overcome those risks
- **Approvals**:
- **Steps for preparing a test plan:**

- Analyze the product (learn product thoroughly)
- Develop test strategy -define scope of testing ,risk and issues
- Define objective of test
- Define test criteria
- Planning the resources
- Plan test environment
- Schedule and cost
- Test deliverables

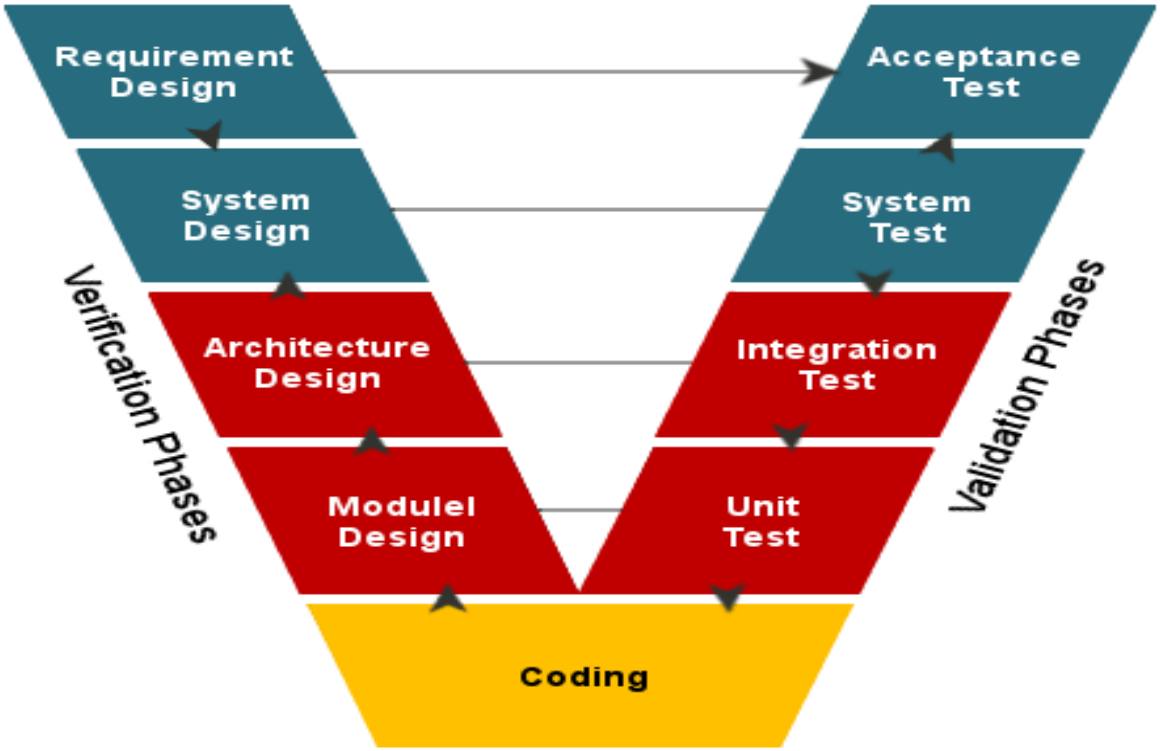
- Advantages of Test Plan:

- Serves as a **guide to testing** throughout the development.
- We **only need to define test points** during the testing phase.
- Serves as **a valuable record** of what testing is done.
- Entire test plan can be **reused** if regression testing is to be done later on.
- Test plan itself could have **defect.** (test the test plan)
-

16) State any four limitations of manual testing

1. Manual testing is slow and costly.
2. It is very labour intensive; it takes a long time to complete tests.
3. Manual tests don't scale well. As the complexity of the software increases the complexity of the testing problem grows exponentially. This leads to an increase in total time devoted to testing as well as total cost of testing.
4. Manual testing is not consistent or repeatable. Variations in how the tests are performed as inevitable, for various reasons. One tester may approach and perform a certain test differently from another, resulting in different results on the same test, because the tests are not being performed identically.
5. Lack of training is the common problem.
6. GUI objects size difference and color combinations are not easy to find in manual testing.
7. Not suitable for large scale projects and time bound projects.
8. Batch testing is not possible, for each test execution Human user interaction is mandatory.
9. Comparing large amounts of data is impractical.
    - 10. Processing change requests during software maintenance takes more time.

17) Describe V-model with labelled diagram.



- **V- Model:**
- It is an extension of waterfall model.
- The process **steps are bent upwards** after coding phase to for V shape.
- V-model demonstrates relationship between each phase of development life cycle and its associate phase of testing.
- **Model is divided into 2 phases:**
  - **verification and**
  - **validation.**
- **Verification:**

i) Requirement analysis:

- First stage.
- **Requirements** of the system are **collected** by analyzing needs of users.
- Users are **interviewed** and **requirements document** is generated.
- Requirement document describes requirements like – **functional ,interface ,performance, data ,security** etc.
- Users do **review** the document.
- **User acceptance tests are designed in this phase**
- Ways of gathering requirements: interviewing, questionnaires, document analysis, observation, throw away prototype, use case, static and dynamic views with users.

- **ii) System Design:**
  - Here system engineers analyze and understand **business** of proposed system by studying user requirement document.
  - They figure out **techniques for implementation**.
  - If any requirement is **not feasible->**user is informed and solution is found->document is edited .
  - Software specification is generated.

- **Which contains**-general system organization, menu structure, data structures etc. and business scenarios, sample windows, reports for understanding.
- Entry diagram and data dictionary is also prepared here.
- **The document for system testing is prepared here.**

- **iii) Architecture Design:**
  - This phase is of **design of computer architecture** and **software architecture**.
  - This is high level design.
  - **It consists of** -List of modules, brief functionality of each module, their interface relationship, dependencies, database tables, architecture diagrams, technology details etc.
  - **Integration testing design is carried out in this phase.**

- **iv) Modular Design:**
  - Also referred as **low level design**.
  - System is divided into **smaller units or modules**.
  - Each module is explained so that **programmer can start coding directly.**
  - **It contains**: detailed functional logic of module in pseudo code, database tables(with elements ,their types and sizes),all interface details with references, all dependency issues, error message listing, complete input and output for a module,
  - **Unit test design is developed in this stage.**

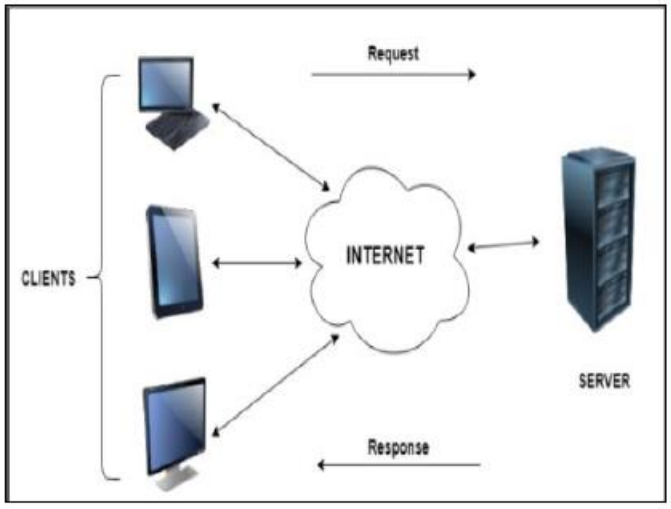
2) Validation:

- **i) Unit Testing:**
  - UTP-unit testing plans are developed during module **design phase.**
  - UTP's are executed **to eliminate bugs** at **code level/unit level.**
  - **Smallest entity** existing independently is called as Unit(a program, module).
  - Unit testing **verifies** that the smallest entity **can function** correctly **when isolated** from rest of the codes/units.
- **ii)Integration Testing:**
  - Integration Test Plans are developed during the **Architectural Design Phase.**
  - These tests verify units created and tested coexist and communicate properly.
  - Test **results** are **shared with customers team**.
- **iii) System Testing:**
  - **System test plans** are developed in this phase.
  - These plans are **composed by clients business team**.
  - **Whole application** is tested for its functionality, interdependency and communication.
  - **Subsets of system testing-** load and performance testing, stress testing, regression testing, etc.
- **iv)User Acceptance Testing:**
  - **UAT- plans are developed** during requirement analysis phase.
  - Test plans are **composed by business users**.
  - Tests are performed in **user environment** that resembles production environment using **realistic data**.
  - UAT checks weather system is ready for use in real time.

- **Advantages of V-model:**
  - **Simple** and **easy to use**.
  - Planning and test design occurs **prior to coding**, it **avoids wastage of time**.
  - Better success than waterfall model.
  - Diagnosis of defects is done in early stages
  - Avoids downward flow of defects.
- This is good for **small projects** where requirements are well
- **Disadvantages of V-model:**
  - This model is **very rigid** and **less flexible**
  - **Development** of s/w is done in **implementation phase**
  - Hence no early prototypes are produced
  - In case of **midway design changes** then we need to update test document as well
  - understood.

18) Explain client-server testing with suitable diagram.

The Client-Server application consists of two systems, one is the Client, and the other is the Server. Here, the client and server interact with each other over the computer network.
In Client-Server application testing, the client sends requests to the server for specific information and the server sends the response back to the client with the requested information. Hence, this testing is also known as two-tier application testing



-

Testing approaches of client server system is as follows.

Component Testing

➢For testing Client and server individually approach and test plan need to be defined.

➢ One may have to devise simulators to replace corresponding components while testing the component targeted by the test.

➢ When server is tested, we may need a client simulator, while testing of client may need server simulator.

Integration Testing

➢After successful testing of servers, clients, and network, they are brought together to form the system and system test cases are executed.

- ➢ Communication between client and server is tested in integration testing.

Performance Testing

➢ System performance is tested when number of clients are communicating with server at a time.

➢ we can test the system under maximum load as well as normal load expected.

Concurrency Testing

➢ It may be possible that multiple users may be accessing same record at a time.

➢ Concurrency testing is required to understand the behaviour of a system under such circumstances.

Disaster Recovery Testing

➢ When the client and server are communicating with each other, there exists a possibility of breaking of the communication due to various reasons or failure of either client or server or link connecting them.

➢ It may involve testing the scenario of such failure at different points in the system and action taken by the system in each case.

Testing for extended periods

➢ In client server application it may be expected that server is running 24*7 for extended period.

➢ one need to conduct testing over an extended period to understand if service level of network and server deteriorates over time due to some reasons.

Compatibility Testing

➢ Servers may be in different hardware, software or operating environment than the recommended one.

➢ Client may differ significantly from the expected environmental variables.

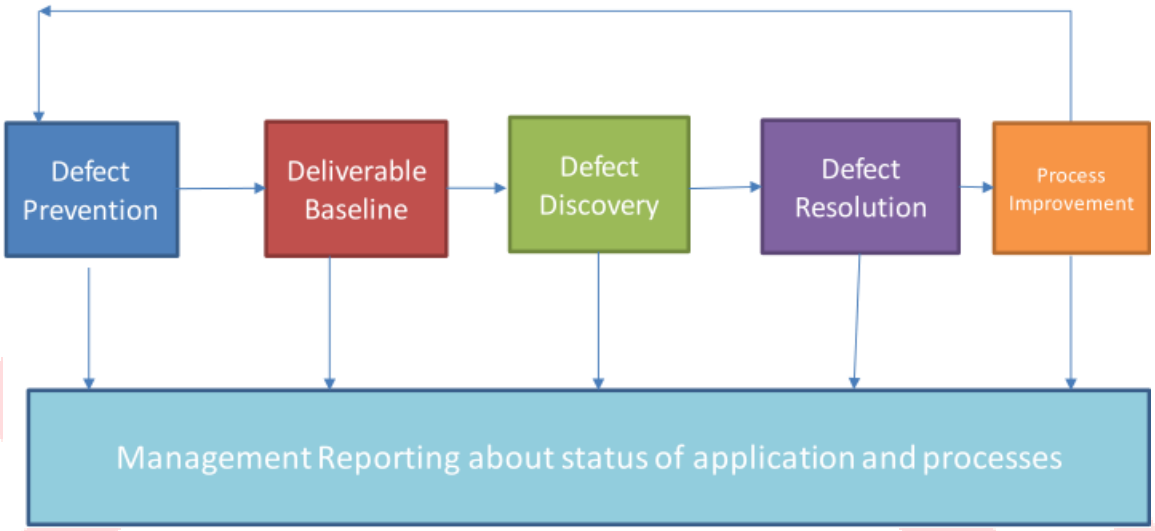
➢ Testing must ensure that performance must be maintained on the range of hardware and software configurations

19) **Explain defect management process with suitable diagram.**

Defect Management Process

Defects found during Verification and validation process in SDLC must be recorded so that it helps in further analysis and root causes of the defect.

The defects found during these phases are used to find weak areas of project /process so that action can be initiated to strengthen it.



-

Defect Management Process: Prevention

- It is a process of improving quality and productivity by preventing the defects into a software product.
- It is virtually impossible to eliminate the defects altogether.
- Implementation of techniques, methodology and standard processes are used to reduce the risk of defects.
- Defect prevention is intended to remove the possibility of any defects before it occurs.

Defect Management Process: Deliverable Baselining

- Once the defect is fixed, retested and found to be closed, the product is created again.
- If the newly created product satisfies the acceptance criteria, it is base lined.
- Only base lined work products can go to the next stage.

Defect Management Process: Defect Discovery

- A defect is said to be discovered when it is brought to the attention of the developers and

acknowledged (i.e., “Accepted”) to be valid one.

- Team should find defects before they become major problems. As soon as a team finds the defects, they should report them so that those can be resolved.

Team should also make sure that defects should be acknowledged by developers and should be valid one.

Defect Management Process: Defect Resolution

Work by the development team to prioritize, schedule and fix a defect, and document the resolution.
This also includes notification back to the tester to ensure that the resolution is verified.

Defect Management Process: Process Improvement

All problems are due to failures in the process involved in creating software.
Defects give an opportunity to identify the problem with process used and update them.
Better processes mean better products with less defect.

Defect Management Process: Management Reporting

Analysis and reporting of defect information to assist management with risk management, process improvement and project management.