Types of testing

I. Unit Testing: Driver, Stub (Question: Explain the concept of stubs and drivers in unit testing. - 6 Marks)

1. Unit is the smallest testable part of the software system.

2. Unit testing is done to verify that the lowest independent entities in any software are working fine.

3. The smallest testable part is isolated from the remainder code and tested to determine whether it works correctly.

4. When developer is coding the software it may happen that the dependent modules are not completed for testing, in such cases developers use stubs and drivers to simulate the called (stub) and caller (driver) units.

5. Unit testing requires stubs and drivers, stubs simulates the called unit and driver simulates the calling unit.

1. STUBS:

i. Assume you have 3 modules, Module A, Module B and module C.

ii. Module A is ready and we need to test it, but module A calls functions from Module B and C which are not ready, so developer will write a dummy module which simulates B and C and returns values to module A.

iii. This dummy module code is known as stub.

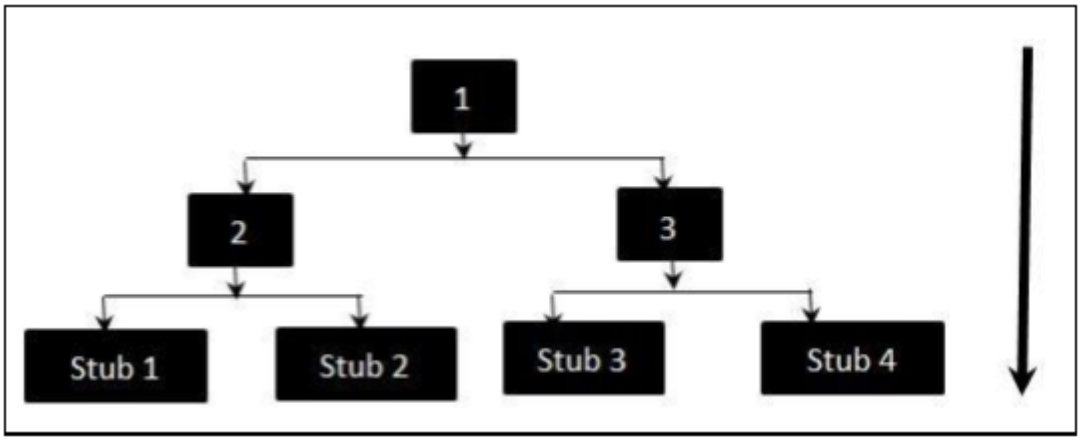

Figure 1: The Stub Flow Diagram

iv. The above diagrams clearly states that Modules 1, 2 and 3 are available for integration, whereas, below modules are still under development that cannot be integrated at this point of time.

v. Hence, Stubs are used to test the modules.

2. DRIVERS:

i. Now suppose you have modules B and C ready but module A which calls functions from module B and C is not ready so developer will write a dummy piece of code for module A which will return values to module B and C.

ii. This dummy piece of code is known as driver.

II. Integration Testing

1. Decomposition Testing: Top Down and Bottom up Integration. (Question: explain top down and bottom up integration with advantages and disadvantages. - 6 Marks)

i. Top down Testing:

In this approach testing is conducted from main module to sub module.

ii. If the sub module is not developed a temporary program called STUB is used for simulate the sub module.

Advantages:

- Advantageous if major flaws occur toward the top of the program.
- Once the I/O functions are added, representation of test cases is easier.
- Early skeletal Program allows demonstrations and boosts morale.

Disadvantages:

- Stub modules must be produced
- Stub Modules are often more complicated than they first appear to be.
- Before the I/O functions are added, representation of test cases in stubs can be difficult.
- Test conditions may be impossible, or very difficult, to create.
- Observation of test output is more difficult. 豈 Allows one to think that design and testing can be overlapped.
- Induces one to defer completion of the testing of certain modules.

i. Bottom up testing:

In this approach testing is conducted from sub module to main module, if the main module is not developed a temporary program called DRIVERS is used to simulate the main module.

Advantages:

- Advantageous if major flaws occur toward the bottom of the program.
- Test conditions are easier to create. 岂 Observation of test results is easier.

Disadvantages:

- Driver Modules must be produced.
- The program as an entity does not exist until the last module is added.

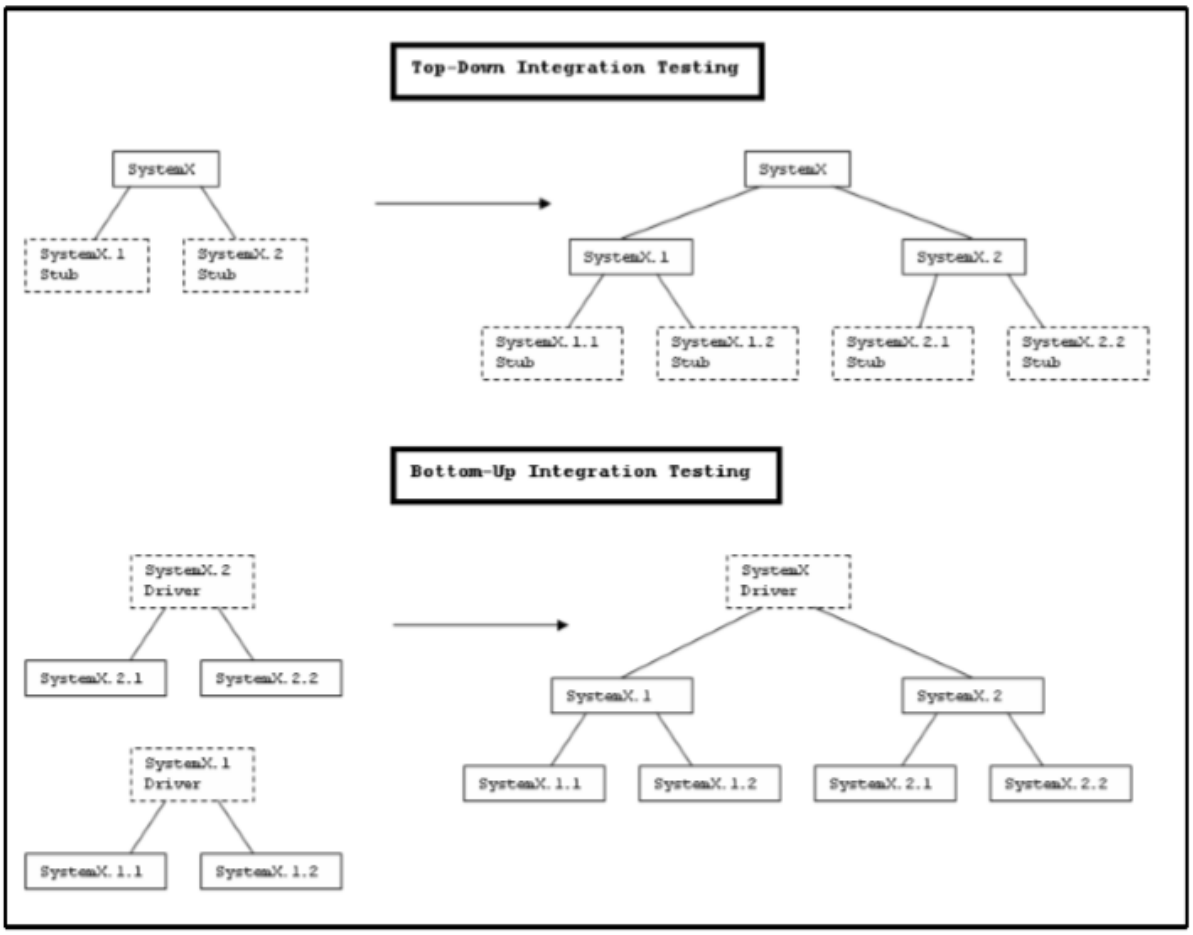

Figure 2: Integration Diagram

2. Bi-Directional Integration (Question: Explain bi directional integration. - 4 Marks)

i. Bi-directional Integration, is a kind of integration testing process that combines top -down and bottom-up testing.

ii. With an experience in delivering Bi-directional testing projects custom software

development services provide the best quality of the deliverables right from the development of software process. iii. Bi-directional Integration testing is a vertical incremental testing strategy that tests the bottom layers and top layers and tests the integrated system in the computer software development process.

iv. Using stubs, it tests the user interface in isolation as well as tests the very lowest level functions using drivers.

v. Bi-directional Integration testing combines bottom-up and top-down testing.

vi. Bottom-up testing is a process where lower level modules are integrated and then tested.

vii. This process is repeated until the component of the top of the hierarchy is analysed. It helps custom software development services find bugs easily without any problems.

viii. Top down testing is a process where the top integrated modules are tested and the procedure is continued till the end of the related module.

ix. Top down testing helps developers find the missing branch link easily.

3. Incremental Integration. (Question: Explain the features of incremental integration. - 4 Marks)

i. After unit testing is completed, developer performs integration testing.

ii. It is the process of verifying the interfaces and interaction between modules.

iii. While integrating, there are lots of techniques used by developers and one of them is the incremental approach.

iv. In Incremental integration testing, the developers integrate the modules one by one using stubs or drivers to uncover the defects.

v. This approach is known as incremental integration testing.

vi. To the contrary, big bang is one other integration testing technique, where all the modules are integrated in one shot.

Features

i. Each Module provides a definitive role to play in the project/product structure

ii. Each Module has clearly defined dependencies some of which can be known only at the runtime. iii. The incremental integration testing's greater advantage is that the defects are found early in a smaller assembly when it is relatively easy to detect the root cause of the same.

iv. A disadvantage is that it can be time-consuming since stubs and drivers have to be developed for performing these tests.

4. Non- Incremental Integration. (Question: Explain the non-incremental integration technique or the big band testing technique. – 4 Marks)

i. The non-incremental approach is also known as "Big-Bang" Testing.

ii. Big Bang Integration Testing is an integration testing strategy wherein all units are linked at once, resulting in a complete system.

iii. When this type of testing strategy is adopted, it is difficult to isolate any errors found, because attention is not paid to verifying the interfaces across individual units.

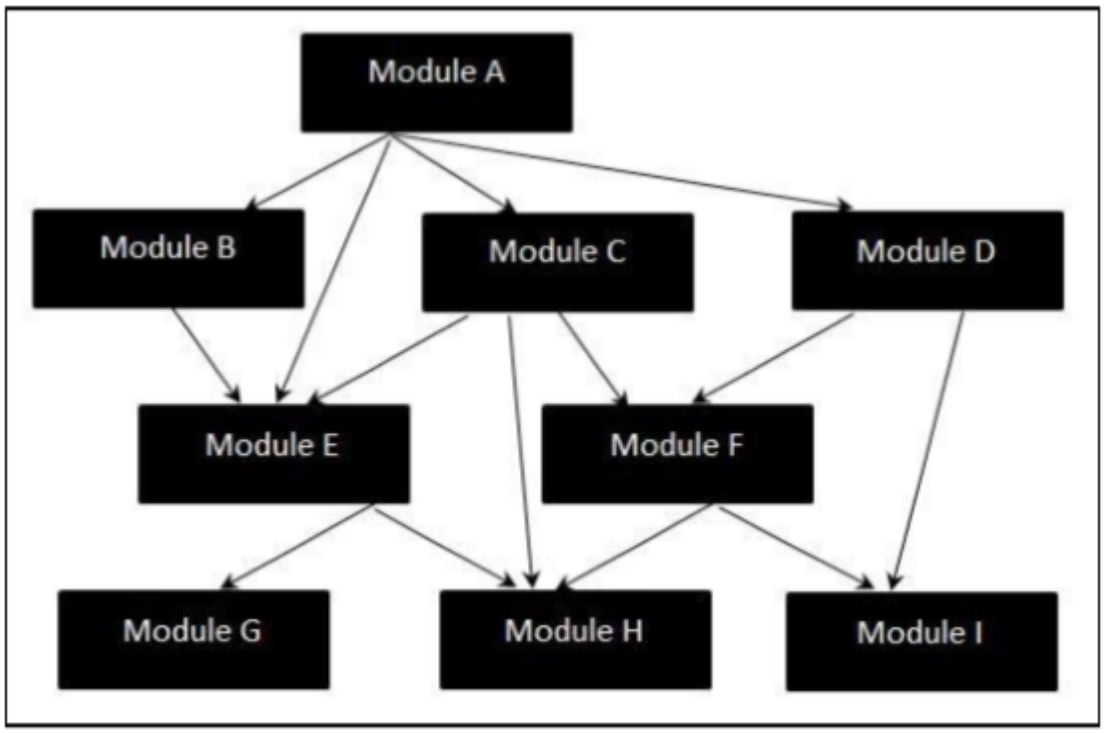

Figure 3: Non-Incremental Integration

Disadvantages

i. Defects present at the interfaces of components are identified at very late stage as all components are integrated in one shot.

ii. It is very difficult to isolate the defects found.

iii. There is high probability of missing some critical defects, which might pop up in the production environment.

iv. It is very difficult to cover all the cases for integration testing without missing

even a single scenario.

III. System Testing (Question: List the various system testing approaches and explain any two. – 8 Marks)

1. System Testing (ST) is a black box testing technique performed to evaluate the complete system the system's compliance against specified requirements.

2. In System testing, the functionalities of the system are tested from an end-to-end perspective.

3. System Testing is usually carried out by a team that is independent of the development team in order to measure the quality of the system unbiased.

4. It includes both functional and Non-Functional testing.

1. Recovery Testing

i. Recovery testing is a type of non-functional testing technique performed in order to determine how quickly the system can recover after it has gone through system crash or hardware failure.

ii. Recovery testing is the forced failure of the software to verify if the recovery is successful.

Steps:

- Determining the feasibility of the recovery process.
- Verification of the backup facilities.
- Ensuring proper steps are documented to verify the compatibility of backup facilities.
- Providing Training within the team.
- Demonstrating the ability of the organization to recover from all critical failures.
- Maintaining and updating the recovery plan at regular intervals.

2. Security Testing

i. Security testing is a testing technique to determine if an information system protects data and maintains functionality as intended.

ii. It also aims at verifying 6 basic principles as listed below:

- Confidentiality
- Integrity
- Authentication
- Authorization
- Availability
- Non-repudiation

Techniques:

- Injection
- Broken Authentication and Session Management
- Cross-Site Scripting (XSS)
- Insecure Direct Object References
- Security Misconfiguration
- Sensitive Data Exposure
- Missing Function Level Access Control
- Cross-Site Request Forgery (CSRF)
- Using Components with Known Vulnerabilities
- Invalidated Redirects and Forwards

3. Performance Testing.

i. Performance testing, a non-functional testing technique performed to determine the system parameters in terms of responsiveness and stability under various workload.

ii. Performance testing measures the quality attributes of the system, such as scalability, reliability and resource usage.

Techniques:

- Load testing - It is the simplest form of testing conducted to understand the behaviour of the system under a specific load. Load testing will result in

measuring important business critical transactions and load on the database, application server, etc., are also monitored.

- Stress testing - 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.
- Soak testing - Soak Testing also known as endurance testing, is performed to determine the system parameters under continuous expected load. During soak tests the parameters such as memory utilization is monitored to detect memory leaks or other performance issues. The main aim is to discover the system's performance under sustained use.
- Spike testing - Spike testing is performed by increasing the number of users suddenly by a very large amount and measuring the performance of the system. The main aim is to determine whether the system will be able to sustain the workload.
- Performance Testing Process:

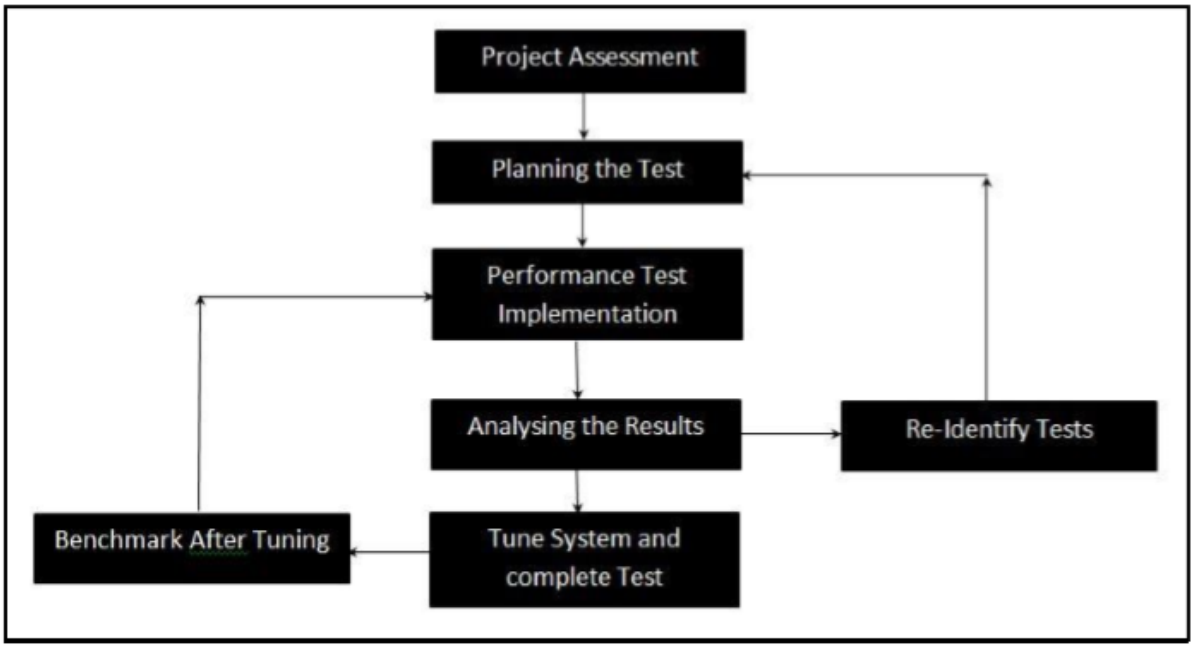

Figure 4: Performance Testing Process

Attributes of Performance Testing:

- Speed
- Scalability

- Stability
- Reliability

4. Load Testing.

i. Load testing is performance testing technique using which the response of the system is measured under various load conditions.

ii. The load testing is performed for normal and peak load conditions.

Load Testing Approach:

- Evaluate performance acceptance criteria
- Identify critical scenarios
- Design workload Model
- Identify the target load levels
- Design the tests
- Execute Tests
- Analyse the Results

Objectives of Load Testing:

- Response time
- Throughput
- Resource utilization
- Maximum user load
- Business-related metrics

5. Stress Testing.

i. Stress testing a Non-Functional testing technique that is performed as part of performance testing. ii. During stress testing, the system is monitored after subjecting the system to overload to ensure that the system can sustain the stress.

iii. The recovery of the system from such phase (after stress) is very critical as it is highly likely to happen in production environment.

Reasons for conducting Stress Testing:

- It allows the test team to monitor system performance during failures.
- To verify if the system has saved the data before crashing or NOT.
- To verify if the system prints meaning error messages while crashing or did it print some random exceptions.
- To verify if unexpected failures do not cause security issues.

Stress Testing - Scenarios:

- Monitor the system behaviour when maximum number of users logged in at the same time.
- All users performing the critical operations at the same time.
- All users accessing the same file at the same time.
- Hardware issues such as database server down or some of the servers in a server park crashed.

6. Usability Testing.

i. Usability testing, a non-functional testing technique that is a measure of how easily the system can be used by end users.

ii. It is difficult to evaluate and measure but can be evaluated based on the below parameters:

- Level of Skill required to learn/use the software. It should maintain the balance for both novice and expert user.
- Time required to get used to in using the software.
- The measure of increase in user productivity if any.
- Assessment of a user's attitude towards using the software.
- Usability Testing Process:

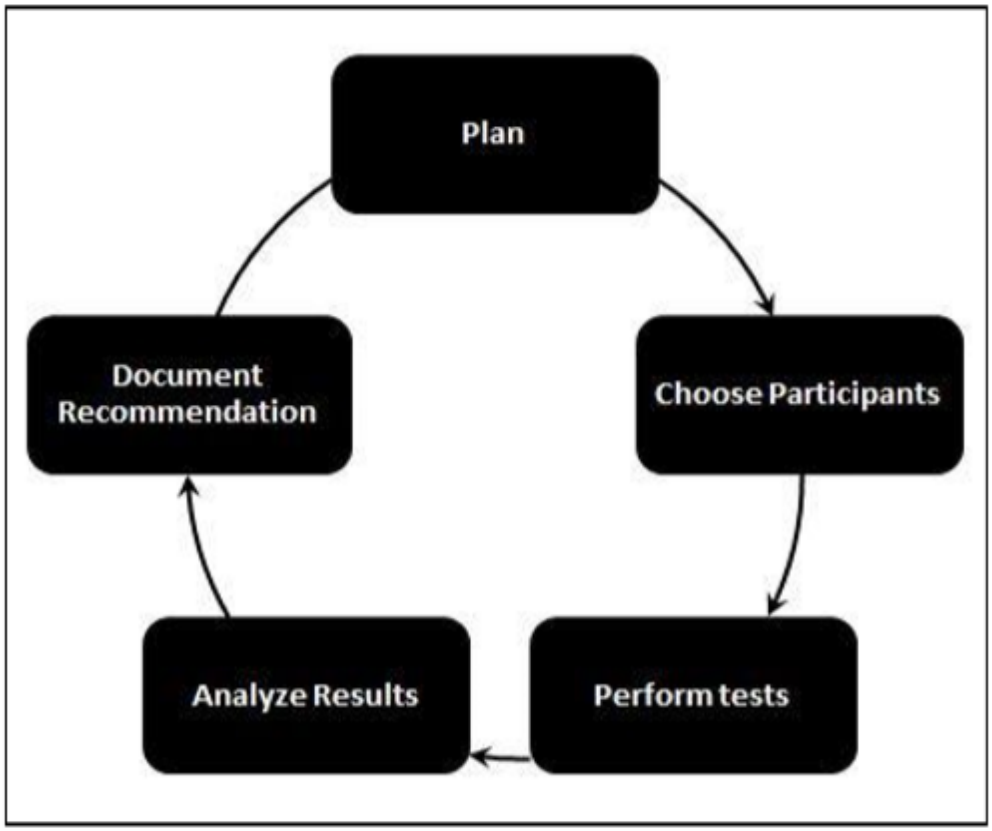

Figure 5: Usability Testing Process

7. Compatibility Testing.

i. Compatibility testing is a non-functional testing conducted on the application to evaluate the application's compatibility within different environments.

ii. It can be of two types - forward compatibility testing and backward compatibility testing.

- Operating system Compatibility Testing - Linux , Mac OS, Windows
- Database Compatibility Testing - Oracle SQL Server
- Browser Compatibility Testing - IE , Chrome, Firefox
- Other System Software - Web server, networking/ messaging tool, etc.

IV. Acceptance Testing.

1. Acceptance Criteria. (Question: Explain the criteria for acceptance testing. - 4 Marks)

i. Comparison testing comprises of comparing the contents of files,

databases, against actual results.

ii. They are capable of highlighting the differences between expected and actual results.

iii. Comparison test tools often have functions that allow specified sections of the files be ignored or masked out.

iv. This enables the tester to mask out the date or time stamp on a screen or field as it is always different from the expected ones when a comparison is performed.

2. Alpha Testing. (Question: Explain alpha testing. - 4 Marks)

Alpha testing takes place at the developer's site by the internal teams, before release to external customers. This testing is performed without the involvement of the development teams.

i. Alpha Testing - In SDLC

The following diagram explains the fitment of Alpha testing in the software development life cycle.

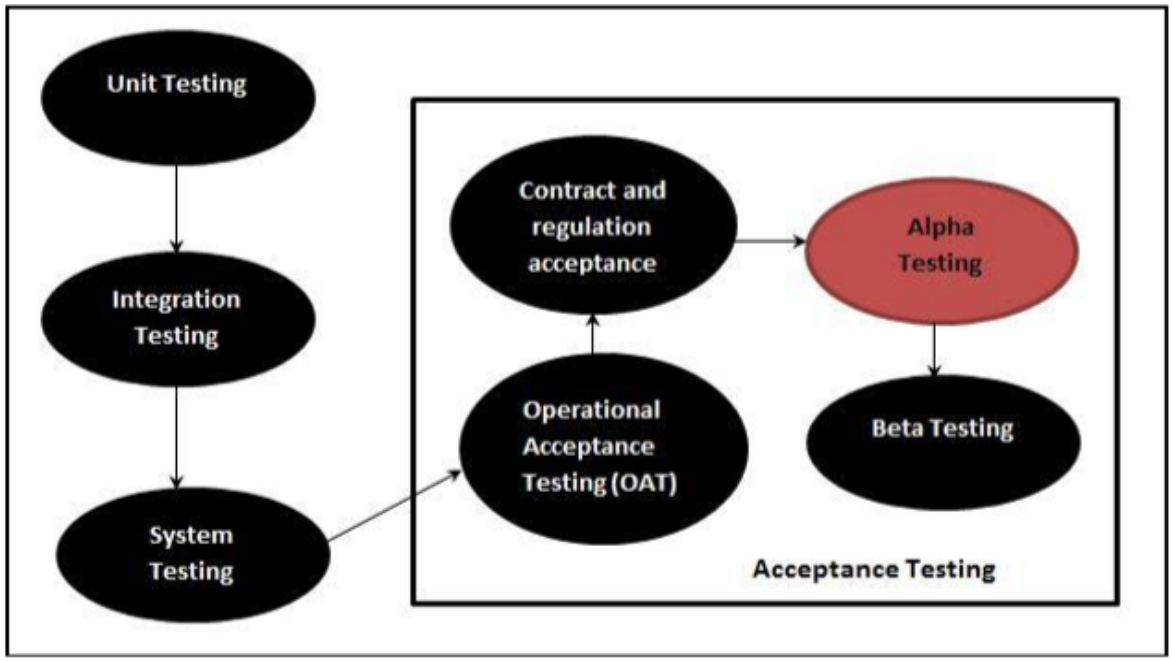

Figure 6: Alpha Testing – SDLC

ii. How do we run it?

- In the first phase of alpha testing, the software is tested by in-house developers during which the goal is to catch bugs quickly.

- In the second phase of alpha testing, the software is given to the software QA team for additional testing.
- Alpha testing is often performed for Commercial off-the-shelf software (COTS) as a form of internal acceptance testing, before the beta testing is performed.

3. Beta Testing. (Question: Explain Beta testing- 4Marks)

i. Beta testing also known as user testing takes place at the end users site by the end users to validate the usability, functionality, compatibility, and reliability testing.

ii. Beta testing adds value to the software development life cycle as it allows the "real" customer an opportunity to provide inputs into the design, functionality, and usability of a product. These inputs are not only critical to the success of the product but also an investment into future products when the gathered data is managed effectively.

Beta Testing - In SDLC

The following diagram explains the fitment of Beta testing in the software development life cycle:

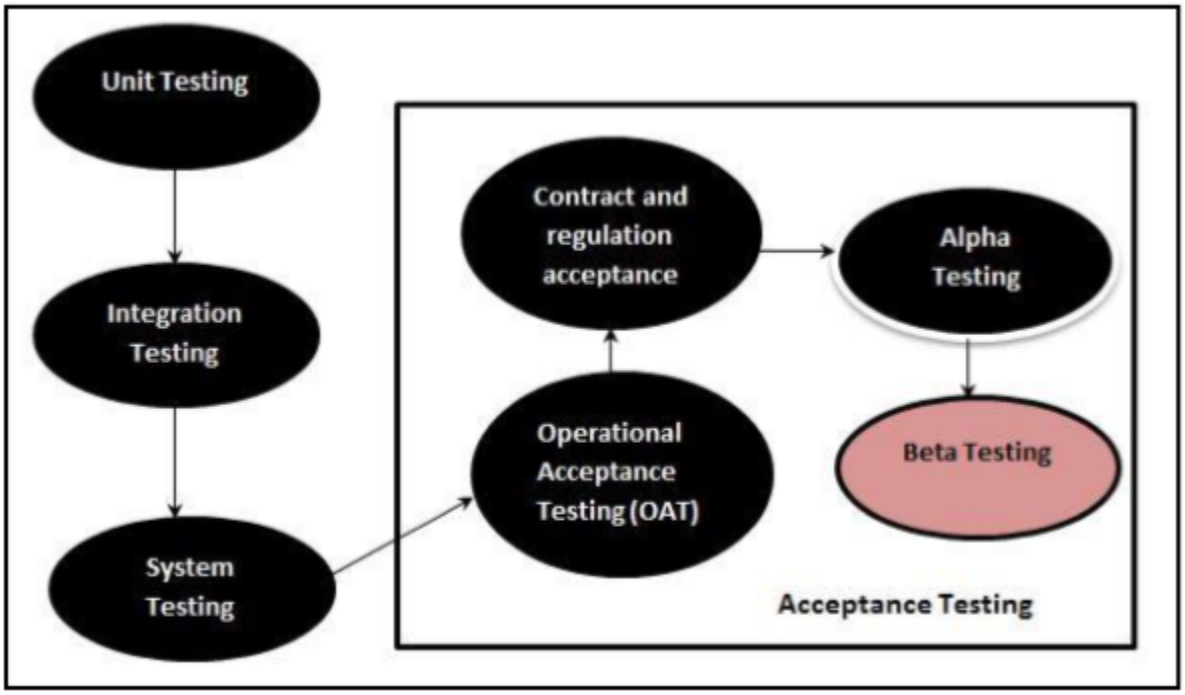

Figure 7: Beta Testing SDLC

V. Special Tests

1. Smoke Testing and Sanity Testing (Question: Differentiate smoke testing and sanity testing. – 6 Marks)

i. Smoke Testing is a testing technique that is inspired from hardware testing, which checks for the smoke from the hardware components once the hardware's power is switched on.

ii. In Software testing context, smoke testing refers to testing the basic functionality of the build.

iii. If the Test fails, build is declared as unstable and it is NOT tested anymore until the smoke test of the build passes.

Smoke Testing - Features:

i. Identifying the business critical functionalities that a product must satisfy.

ii. Designing and executing the basic functionalities of the application.

iii. Ensuring that the smoke test passes each and every build in order to proceed with the testing.

iv. Smoke Tests enables uncovering obvious errors which saves time and effort of test team. v. Smoke Tests can be manual or automated.

i. **Sanity testing**, a software testing technique performed by the test team for some basic tests. The aim of basic test is to be conducted whenever a new build is received for testing. The terminologies such as Smoke Test or Build Verification Test or Basic Acceptance Test or Sanity Test are interchangeably used, however, each one of them is used under a slightly different scenario.

ii. Sanity test is usually unscripted, helps to identify the dependent missing functionalities. It is used to determine if the section of the application is still working after a minor change. iii. Sanity testing can be narrow and deep. Sanity test is a narrow regression test that focuses on one or a few areas of functionality.

2. Regression Testing. (Question: Explain Regression Testing. – 4 Marks)

i. Regression testing a black box testing technique that consists of re-executing those tests that are impacted by the code changes.

ii. These tests should be executed as often as possible throughout the

software development life cycle.

Types of Regression Tests:

i. Final Regression Tests: - A "final regression testing" is performed to validate the build that hasn't changed for a period of time. This build is deployed or shipped to customers.

ii. Regression Tests: - A normal regression testing is performed to verify if the build has NOT broken any other parts of the application by the recent code changes for defect fixing or for enhancement.

3. Usability Testing. (Question: Explain Usability Testing. – 4 Marks)

i. Usability testing, a non-functional testing technique that is a measure of how easily the system can be used by end users.

ii. It is difficult to evaluate and measure but can be evaluated based on the below parameters:

iii. Level of Skill required to learn/use the software. It should maintain the balance for both novice and expert user.

iv. Time required to get used to in using the software.

v. The measure of increase in user productivity if any.

vi. Assessment of a user's attitude towards using the software. vii. The usability process is shown in Figure

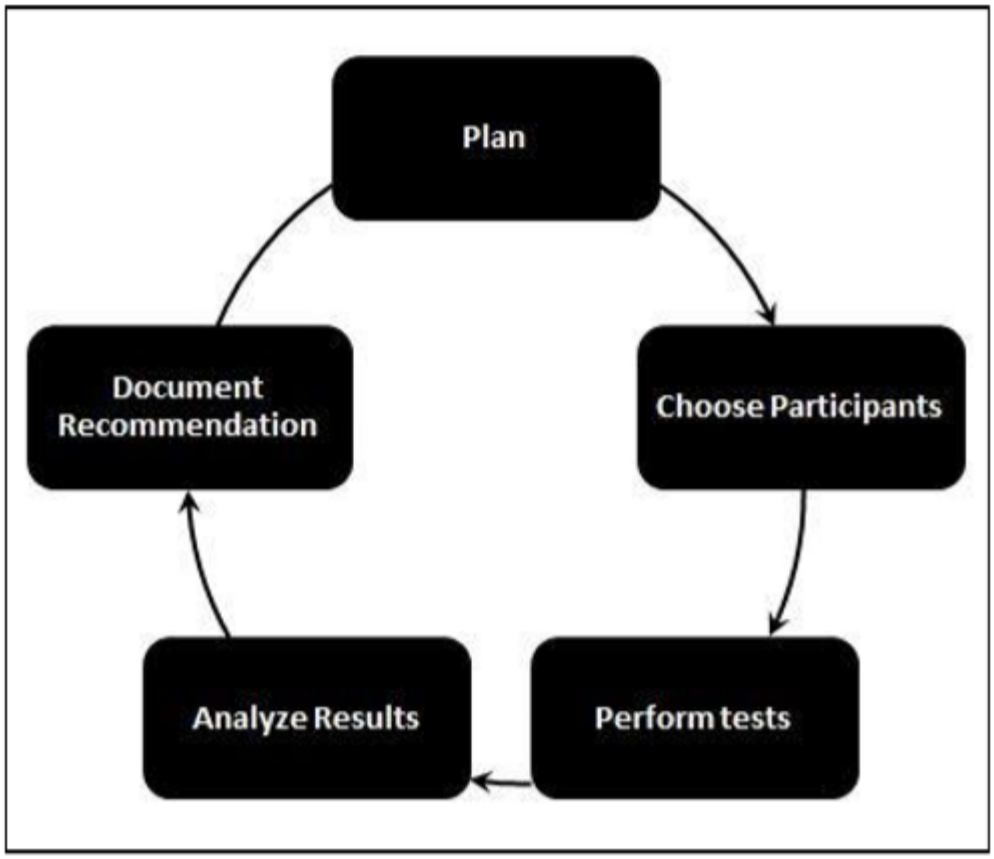

Figure 8: Usability Testing

4. GUI Testing. (Question: Explain GUI Testing. – 4 Marks)

i. GUI testing is a testing technique in which the application's user interface is tested whether the application performs as expected with respect to user interface behaviour.

ii. GUI Testing includes the application behaviour towards keyboard and mouse movements and how different GUI objects such as toolbars, buttons, menu bars, dialog boxes, edit fields, lists, behaviour to the user input.

GUI Testing Guidelines

i. Check Screen Validations

ii. Verify All Navigations

iii. Check usability Conditions

iv. Verify Data Integrity

v. Verify the object states

vi. Verify the date Field and Numeric Field Formats GUI Automation Tools

Following are some of the open source GUI automation tools in the market:

Product	Licensed Under	URL
AutoHotkey	GPL	http://www.autohotkey.com/
Selenium	Apache	http://docs.seleniumhq.org/
Sikuli	MIT	http://sikuli.org
Robot Framework	Apache	www.robotframework.org
Water	BSD	http://www.watir.com/
Dojo Toolkit	BSD	http://dojotoolkit.org/

Table 1:GUI Automation

Following are some of the Commercial GUI automation tools in the market.

Product	Vendor	URL
AutoIT	AutoIT	http://www.autoitscript.com/site/autoit/
EggPlant	TestPlant	www.testplant.com
QTP	Hp	http://www8.hp.com/us/en/software-solutions/
Rational Functional Tester	IBM	http://www-03.ibm.com/software/products/us/en/functional
Infragistics	Infragistics	www.infragistics.com
iMacros	iOpus	http://www.iopus.com/iMacros/
CodedUI	Microsoft	http://www.microsoft.com/visualstudio/

Table 2:Commercial GUI Automation

5. Object Oriented Application Testing. (Question: Explain object oriented application testing. - 6 Marks)

i. The Full-Lifecycle Object-Oriented Testing (FLOOT) methodology is a collection of testing techniques to verify and validate object-oriented software.

ii. The FLOOT lifecycle is depicted in Figure 9, indicating a wide variety of techniques (described in Table 9 are available to you throughout all aspects of software development.

iii. The list of techniques is not meant to be complete: instead the goal is to make it explicit that you have a wide range of options available to you.

iv. It is important to understand that although the FLOOT method is presented as a collection of serial phases it does not need to be so: the techniques of FLOOT can be applied with evolutionary/agile processes as well.

v. The reason why I present the FLOOT in a "traditional" manner is to make it explicit that you can in fact test throughout all aspects of software development, not just during coding.

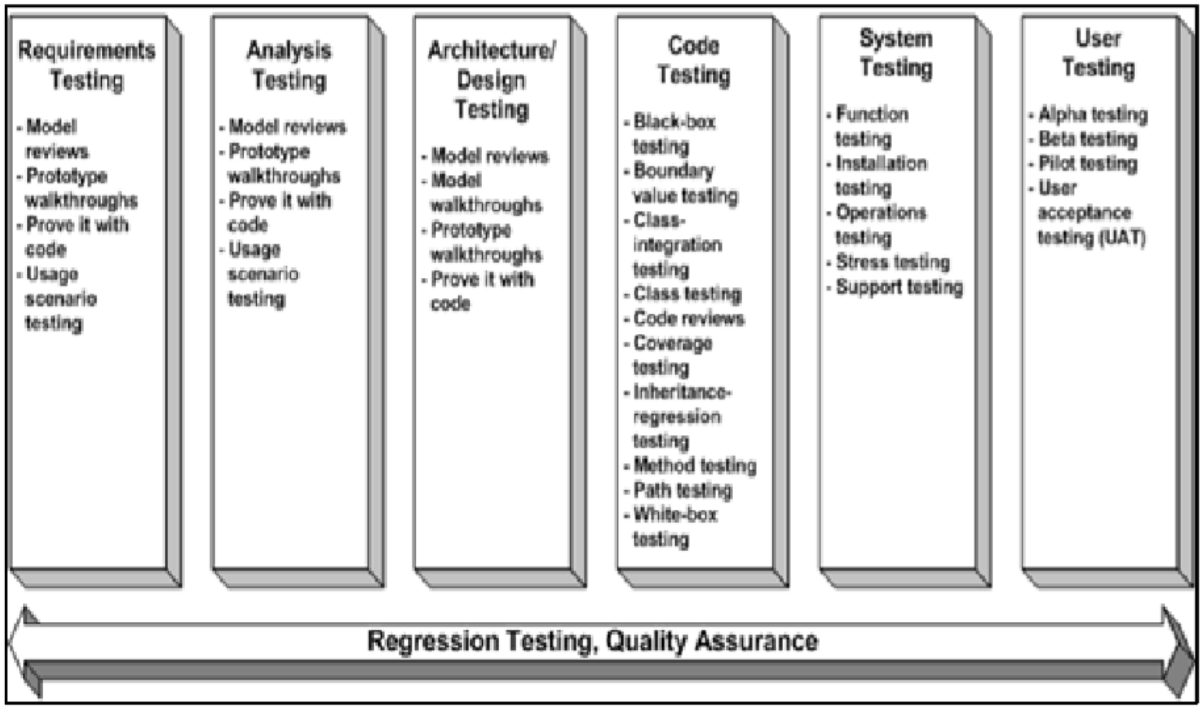

Figure 9: FLOOT Diagram

6. Client Server Testing. (Question: Explain Client Server Testing. – 4 Marks)

i. This type of testing usually done for 2 tier applications (usually developed for LAN) Here we will be having front-end and backend.

ii. The application launched on front-end will be having forms and reports which will be monitoring and manipulating data.E.g: applications developed in VB, VC++, Core Java, C, C++, D2K, PowerBuilder etc.,

iii. The backend for these applications would be MS Access, SQL Server, Oracle, Sybase, Mysql, Quadbase.

iv. The tests performed on these types of applications would be– User interface testing Manual support testing– Functionality testing– Compatibility testing & configuration testing – Intersystem testing.

7. Web Based Testing. (Question: Explain web based testing. - 4 Marks)

i. Web application testing, a software testing technique exclusively adopted to

test the applications that are hosted on web in which the application interfaces and other functionalities are tested.

Web Application Testing - Techniques:

1. Functionality Testing - The below are some of the checks that are performed but not limited to the below list:

- Verify there is no dead page or invalid redirects.
- First check all the validations on each field.
- Wrong inputs to perform negative testing.
- Verify the workflow of the system.
- Verify the data integrity.

2. Usability testing - To verify how the application is easy to use with.

- Test the navigation and controls.
- Content checking.
- Check for user intuition.

3. Interface testing - Performed to verify the interface and the dataflow from one system to other.

4. Compatibility testing- Compatibility testing is performed based on the context of the application.

- Browser compatibility
- Operating system compatibility
- Compatible to various devices like notebook, mobile, etc.

5. Performance testing - Performed to verify the server response time

and throughput under various load conditions.

- Load testing - It is the simplest form of testing conducted to understand the behaviour of the system under a specific load. Load testing will result in measuring important business critical transactions and load on the database, application server, etc. are also monitored.
- Stress testing - 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.
- Soak testing - Soak Testing also known as endurance testing, is performed to determine the system parameters under continuous expected load. During soak tests the parameters such as memory utilization is monitored to detect memory leaks or other performance issues. The main aim is to discover the system's performance under sustained use.
- Spike testing - Spike testing is performed by increasing the number of users suddenly by a very large amount and measuring the performance of the system. The main aim is to determine whether the system will be able to sustain the work load.

6. Security testing - Performed to verify if the application is secured on web as data theft and unauthorized access are more common issues and below are some of the techniques to verify the security level of the system.

- Injection
- Broken Authentication and Session Management
- Cross-Site Scripting (XSS)
- Insecure Direct Object References
- Security Misconfiguration
- Sensitive Data Exposure
- Missing Function Level Access Control
- Cross-Site Request Forgery (CSRF)
- Using Components with Known Vulnerabilities

- Invalidated Redirects and Forwards