Chapter 2 –

Testing Levels & Types

- Mrs. Apurwa Barve

Stubs and Drivers

- ✓ **Stubs and drivers** are used during unit and integration testing to substitute for missing or incomplete parts of a system.
- ✓ They allow for the testing of individual components in isolation before they are integrated into the complete application.
- ✓ **Stubs simulate functions that are called (low-level modules),** while **drivers simulate functions that are calling (high-level modules).**
- ✓ Provide predefined responses so the tested module can function as if the dependent modules were complete.
- ✓ Stub :
  - ➢ Used to simulate modules called by the component being tested.
  - ➢ Used when lower-level modules that a component depends on are not ready.
  - ➢ Examples:

  1. Testing a module that retrieves data from a database. The stub would contain predefined data and return that data when specific queries are made, allowing the developer to verify how the tested module processes that data.
  2. Testing a component that interacts with the file system (e.g., reading or writing files).
  3. Testing error handling - Stubs can be programmed to return specific error messages or exceptions when called, allowing you to test your error handling logic without causing actual errors.

Stubs and Drivers

- ✓ Drivers :
  - ➢ Used to simulate modules called by the component being tested.
  - ➢ Used when higher-level modules that call a component are not ready.
  - ➢ Examples:

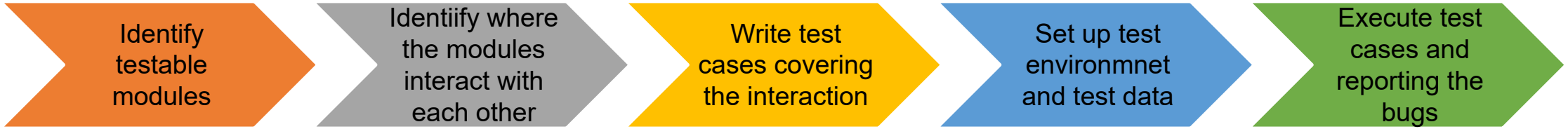
  1. If the actual UI hasn't been built yet, a driver can be created to simulate user input by providing hardcoded data or generating various input sequences (e.g., button clicks, text entries) to the data processing module.

  2. In a Calculator application, writing a driver program implementation which will call different low level modules such as addition, subtraction, multiplication, division, trignometric functions, etc and providing it with hardcoded data to see if these modules are well integrated.

Integration Testing

- ✓ It is performed after unit testing and before system testing.
- ✓ It validates the interactions and interfaces between different modules or components of a software application.
- ✓ The goal is to ensure that these modules work correctly when combined and exchange data seamlessly to achieve the desired functionality.
- ✓ Advantages:
    1. Helps in the early detection of issues that arise from interactions between components.
    2. Improves the overall reliability of the system.
    3. Verifies the correct flow of data between the modules
    4. Contributes to an enhanced user experience by minimizing glitches and disruptions.

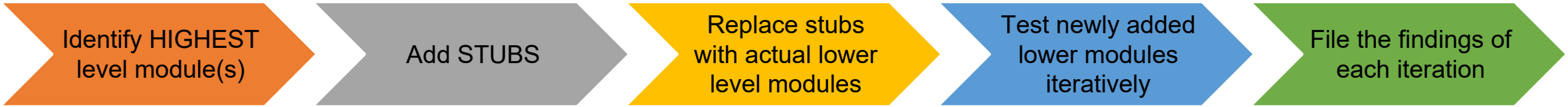
- ✓ Steps to perform Integration testing:



Integration Testing - Approaches

1. Top down appraoch :

- ✓ It starts from the highest-level modules (main components) and progressively moving down to the lower-level modules (sub-components).
- ✓ Steps to perform Integration testing:



Integration Testing - Approaches

2. Bottom up approach :

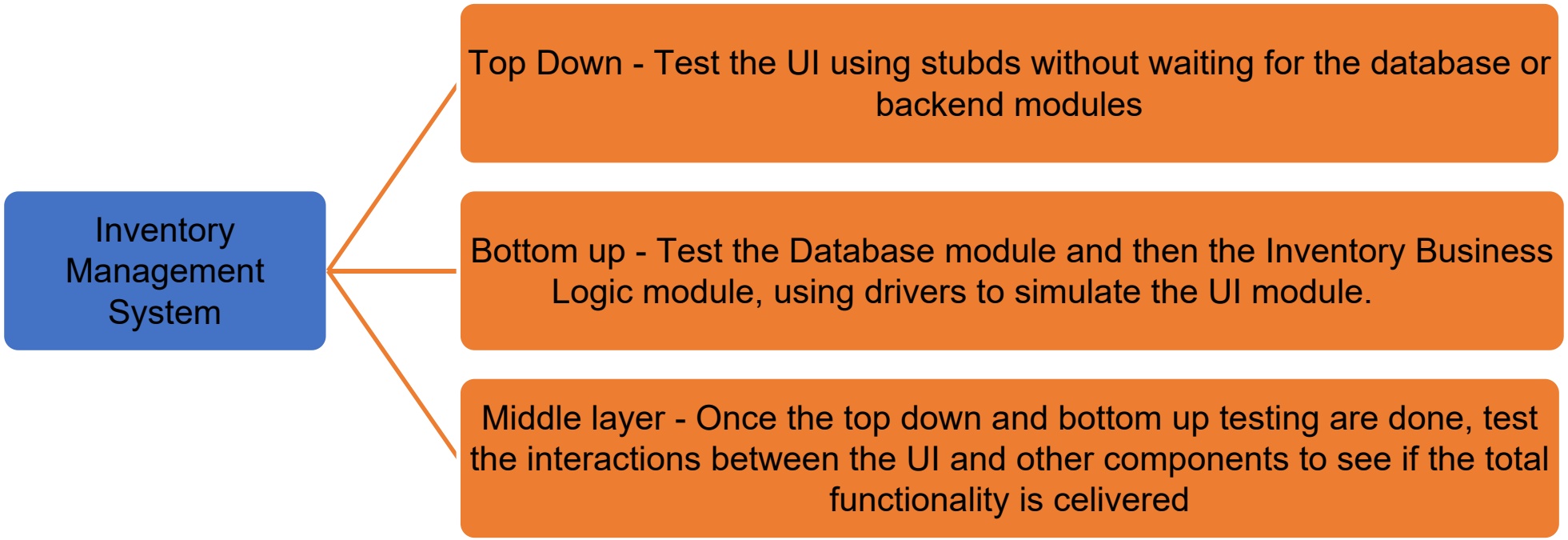
- ✓ It starts testing starts with the lowest-level or sub-modules, and then progressively integrates and tests higher-level or main modules until the entire system is tested as a whole
- ✓ Steps to perform Integration testing:

Activity 2.1

- ✓ Discuss how you will test an e-commerce platform using Top Down intergration appraoch ?
- ✓ Discuss how you will test an e-commerce platform using Bottom up intergration approach?

Integration Testing - Approaches

1. Bidirectional appraoch :
   - ✓ Also called as Sandwich Integration testing.
   - ✓ Combines both top-down and bottom-up integration testing methods, meeting in a middle layer.
   - ✓ This approach is particularly effective for complex systems with multiple layers and allows for parallel development and testing efforts
   - ✓ Consider the following example of Sandwich / Bidirectional testing appraoch



Activity 2.2

- ✓ Discuss how you will test a video streaming platform using Bidirectional intergration appraoch ?

Performance Testing

- ✓ Assesses how well a system or application performs under various workloads and conditions.
- ✓ Main parameters of performance testing of a system are:
  a. **Responsiveness** - The time it takes for a system to respond to a user request. Also, the number of transactions the system can process within a specific period.
  b. **Stability** - The number of failed transactions or errors that occur during the test and how the system behaves to errors
  c. **Scalability / Concurrency** - The number of users or requests the system can handle at the same time.
  d. **CPU Storage** - The amount of processing power being used by the system

Performance Testing Types

1. Load Testing -

- This type of testing evaluates a system's behavior under an expected, real-world user load to ensure it remains stable and responsive
- The goal is to see if the system can handle the typical number of users or transactions it was designed for without performance degradation
- It helps identify performance bottlenecks, such as slow response times or high resource usage, before an application is released.
- Vitual users are generated to test the following parameters of the system.

  a. Respose time

  b. Throughput

  c. Resource utilization

  By gradually increasing the number of virtual users, the test can determine at what point performance begins to decline, helping to pinpoint bottlenecks and identify areas for optimization.

Activity 2.3

- ✓ Discuss how you will test the following sites using Load Performance Testing in the given situations
  - a. e-commerce site such as Amazon before Diwali sale
  - b. Railway booking site before commencement of holiday season.
  - c. AICTE/DTE site during form filling process before commencement of admission rounds.

Performance Testing Types

2. StressTesting -

- This type of testing pushes a system beyond its normal operational limits to see how it behaves under extreme conditions.
- The goal is to find the system's "breaking point," understand how it fails, and, most importantly, how it recovers from that failure.
- While load testing simulates an expected user load, stress testing intentionally creates an abnormal, heavy workload often referred to as TORTURE testing.
- Following parameters of the system are testing using STRESS testing.

  a. Robustness

  b.Stability

  c. Error-handling capabilities.

  d. Data integrity

  e. Graceful degradation

  f. System failure point

Activity 2.4

- ✓ Discuss how you will test the following sites using Stress Performance Testing in the given situations
  - a. Online Ticket Booking platform for a big concert
  - b. Result website

Performance Testing Types

3. SecurityTesting -

- This type of testing is a specialized approach that checks a system's security while it is operating under a heavy workload.
- Security vulnerabilities may not be apparent during normal operation but can be exposed or exploited when the system is strained.
- Following parameters of the system are testing using SECURITY testing.

  a. Resource Depletion - A server running out of memory or CPU can slow down security checks, creating a window for an attack.

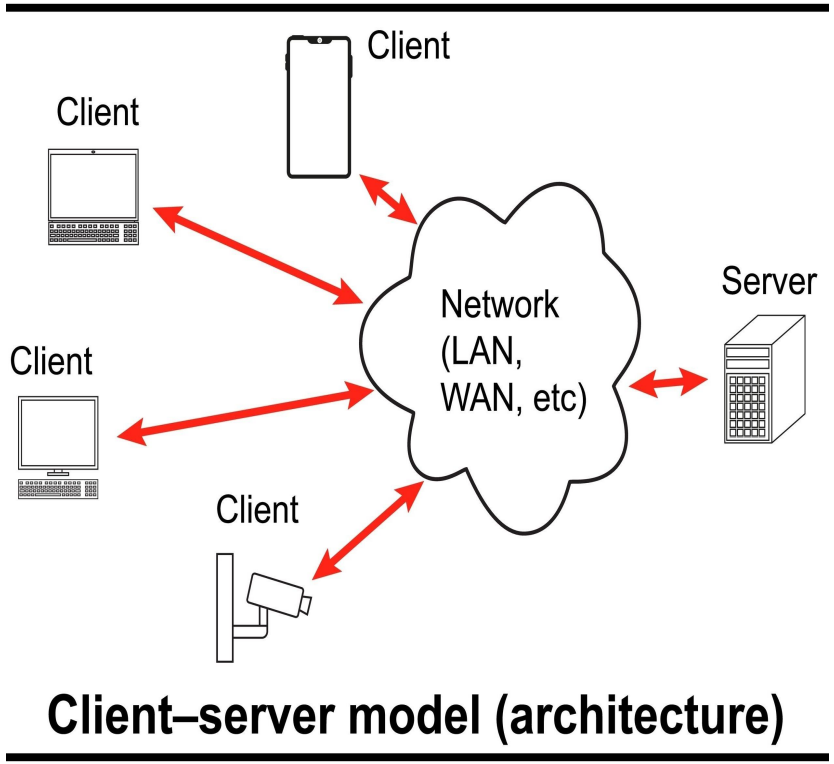
  b. Race condition - When multiple processes or users try to access the same resource simultaneously under heavy load, it can lead to race conditions that bypass security controls.

  c. Error handling failures - Is sensitive information leaked during error handling?

Performance Testing Types

4. Client server Testing -

- This type of testing validates the functionality and performance of a system that is divided into a client (user interface) and a server (back-end).
- The goal is to ensure that the communication and data exchange between the client and server work correctly and securely.



Client–server model (architecture)

Performance Testing Types

4. Client server Testing -

- This type of testing is essential for any application where a user interacts with a front-end interface to access data or services from a remote server, such as a website, a mobile app, or a desktop application connected to a database.

Example - **User using social media app to post a photo on his/her account.**

Solution :

Step 1 - Identify the CLIENT and the SERVER.

Step 2 - Check if following things are working fine :

a. **Functional Testing** - Check if the photo appears correctly in the user's feed after they tap "Post." and if the photo is saved properly on the server's database.

b. **Performance Testing** - The team would simulate a large number of users posting photos at the same time to see if the server can handle the load and if the response time for each user remains fast.

c. **Security Testing** - Check if it is possible to bypass the login process or attempt to send a malicious file to the server instead of an image to see how the system responds. Also verify that the data (the photo and user information) is encrypted during transmission.

Regression Testing

- ✓ This type of testing makes sure a recent change to the code hasn't broken anything that was previously working.
- ✓ Regressions means new bugs which can be introduced in the existing code because of new code changes. This type of testing ensures that the software's existing features continue to function as expected after modifications like bug fixes, new feature additions, or system updates.
- ✓ This is performed -
  - a. After some bug(s) is/are fixed
  - b. After any new feature(s) is/are added to the existing system
  - c. After any performance improvement features are added
  - d. After some major system update or migration

Activity 2.5

✓ Devlopment team has just fixed a bug related to shopping cart functionality where previously the user could not remove a product which was added to the cart. Discuss how you will use regression testing to verify the bug fix.

Solution :

Step 1 - Test the ACTUAL bug.

Step 2 - Test the REST of the functionalities of that flow.

Step 3 - Test the RELATED activities around that flow.

GUI Testing

- ✓ This type of testing that validates the graphical user interface of an application.
- ✓ The goal is to ensure that all the visual elements—like buttons, icons, menus, text boxes, and windows—work correctly and are aesthetically pleasing to the user.
- ✓ It tests the following aspects:-

  a. Functionality - Do the buttons and links do what they are supposed to? For example, clicking a "Submit" button should process a form.

  b. Usability - Is the interface easy to navigate and understand? This includes checking for clear labels, logical layouts, and intuitive workflows.

  c. Layout and Design - Are all elements positioned correctly? Are colors, fonts, and sizes consistent with the design specifications? For instance, does a button fit inside its designated area without being cut off?

  d. **Compatibility -** Does the application's GUI work correctly on different screen sizes, resolutions, operating systems, and browsers? This is crucial for web and mobile applications.

Activity 2.6

- ✓ Discuss any app which is well designed GUI wise.
- ✓ Discuss any app which is NOT well designed GUI wise.

Thank You