Chapter 5: Levels of testing

Objectives

- At the end of this chapter ,the student will be able to achieve:-
  - Unit testing
  - Integration testing
  - System Testing
  - Regression Testing
  - Acceptance testing (Alpha, beta)

Introduction of Level of testing

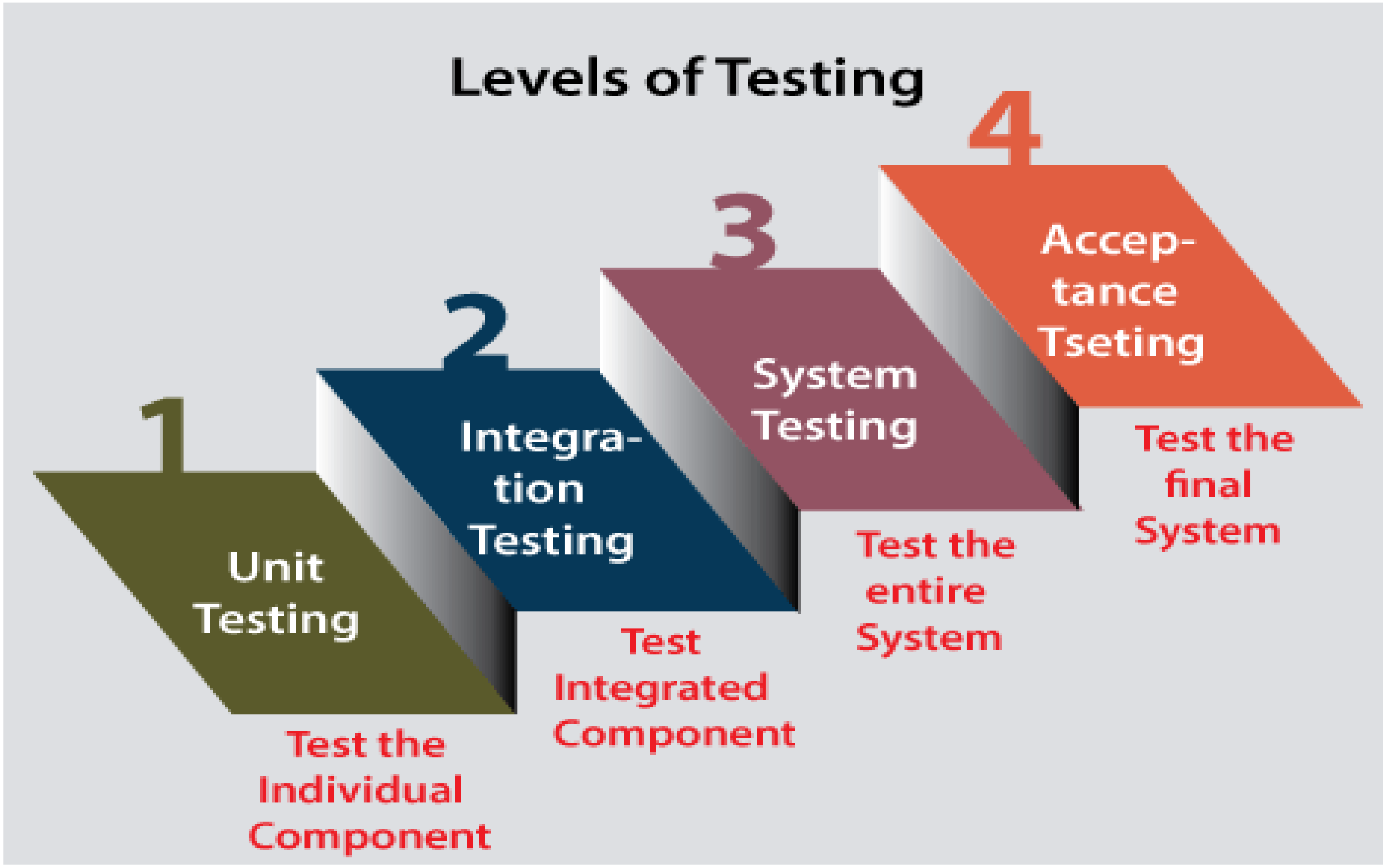
- Testing any application or software, the test engineer needs to follow multiple testing techniques.
- In order to detect an error, we will implement software testing;
  - all the errors can be removed to find a product with more excellent quality.
- Testing levels are the procedure for finding the missing areas and avoiding overlapping and repetition between the development life cycle stages.
  - SDLC (Software Development Life Cycle).

Introduction of Level of testing Cont.

- The levels of software testing involve the different methodologies, which can be used while we are performing the software testing.
- In software testing, we have four different levels of testing, which are as discussed below:

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

Introduction of Level of testing Cont.



Unit Testing

- Unit testing is the first level of software testing, which is used to test if software modules are satisfying the given requirement or not.
- The first level of testing involves analyzing each unit or an individual component of the software application.
- Unit testing is also the first level of functional testing.
- The primary purpose of executing unit testing is to validate unit components with their performance.
- A unit component is an individual function or regulation of the application, or we can say that it is the smallest testable part of the software.

Unit Testing Cont.

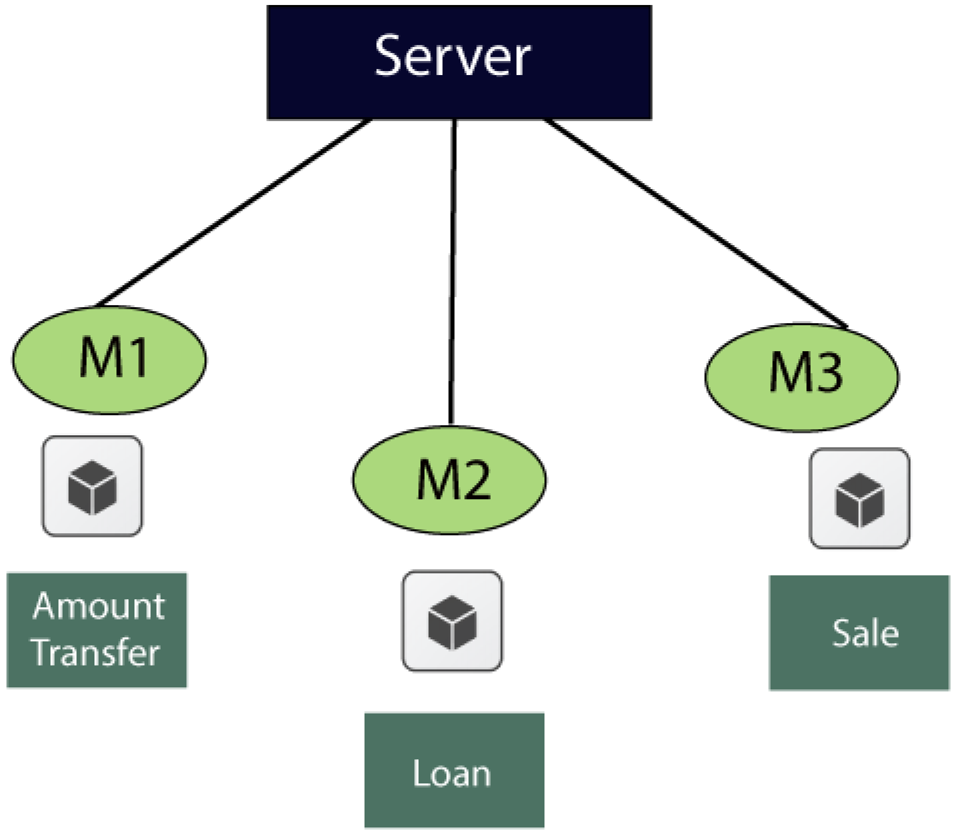
- The reason of performing the unit testing is to test the correctness of inaccessible code.
- The developers implement the unit.
- the application is ready and given to the Test engineer, he/she will start checking every component of the module or module of the application independently or one by one, and this process is known as unit testing or components testing.

Unit Testing Cont.

- Some crucial reasons are listed below:
  - Unit testing helps tester and developers to understand the base of code that makes them able to change defect causing code quickly.
  - Unit testing helps in the documentation.
  - Unit testing fixes defects very early in the development phase that's why there is a possibility to occur a smaller number of defects in upcoming testing levels.
  - It helps with code reusability by migrating code and test cases.

Unit Testing Cont.

- Example of Unit testing



Unit Testing Cont.

For the amount transfer, requirements are as follows:

1.Amount transfer

  - 1.1From account number (FAN)→ Text Box
    - 1.1.1FAN→ accept only 4 digit
  - 1.2 To account no (TAN)→ Text Box
    - 1.2.1TAN→ Accept only 4 digit
  - 1.3Amount→ Text Box
    - 1.3.1Amount → Accept maximum 4 digit
  - 1.4Transfer→ Button
    - 1.4.1 Transfer → Enabled
  - 1.5Cancel→ Button
    - 1.5.1Cancel→ Enabled

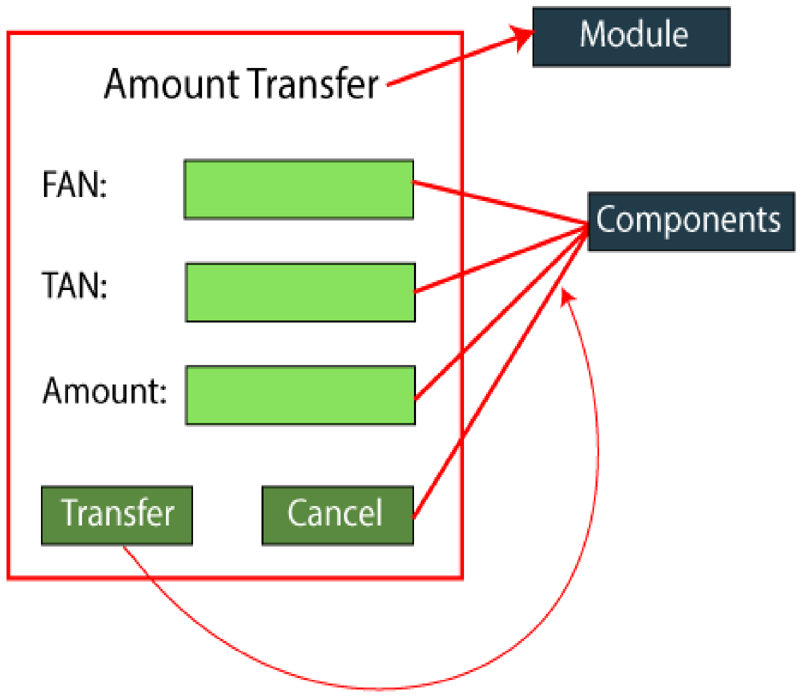
Unit Testing Cont.

- Below are the application access details, which is given by the customer
  - URL→ login Page
  - Username/password/OK → home page
  - To reach Amount transfer module follow the below

  Loans → sales → Amount transfer
- While performing unit testing, we should follow some rules, which are as follows:
  - To start unit testing, at least we should have one module.
  - Test for positive values
  - Test for negative values
  - No over testing
  - No assumption required

Unit Testing Cont.

- When we feel that the maximum test coverage is achieved, we will stop the testing.
- Now, we will start performing the unit testing on the different components such as
  - From account number(FAN)
  - To account number(TAN)
  - Amount
  - Transfer
  - Cancel



Unit Testing Cont.

For the FAN components

Values	Description
1234	accept
blank	Error message→ enter some values
5 digit/ 3 digit	Error message→ accept only 4 digit
Alphanumeric	Error message → accept only digit
Blocked account no	Error message
Copy and paste the value	Error message→ type the value
Same as FAN and TAN	Error message

For the TAN component

- Provide the values just like we did in **From account number** (FAN) components

Unit Testing Cont.

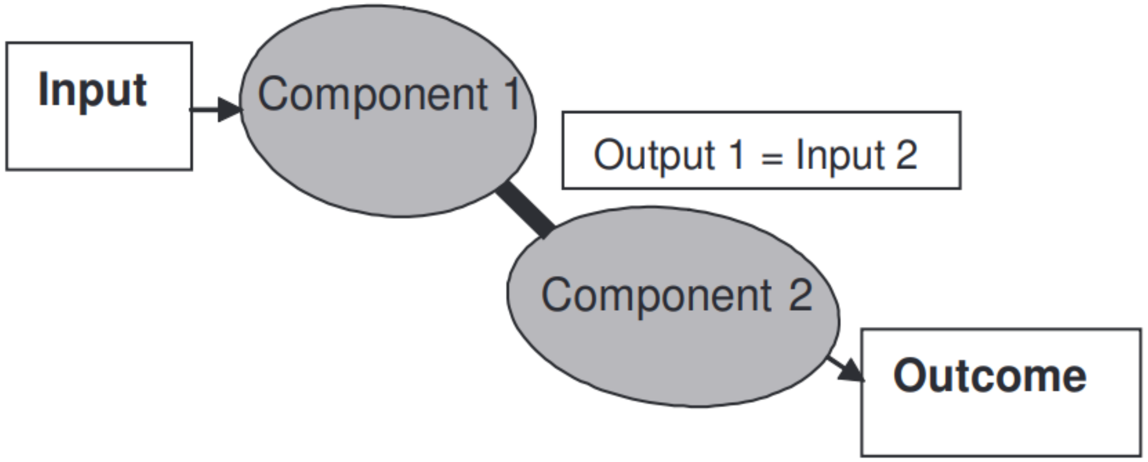
- For Transfer component
  - Enter valid FAN value
  - Enter valid TAN value
  - Enter the correct value of Amount
  - Click on the Transfer button→ amount transfer successfully(confirmation message)
- For Cancel Component
  - Enter the values of FAN, TAN, and amount.
  - Click on the Cancel button → all data should be cleared.

Unit Testing Cont.

- Unit testing uses all white box testing techniques as it uses the code of software application:
  - Data flow Testing
  - Control Flow Testing
  - Branch Coverage Testing
  - Statement Coverage Testing
  - Decision Coverage Testing

Integration Testing

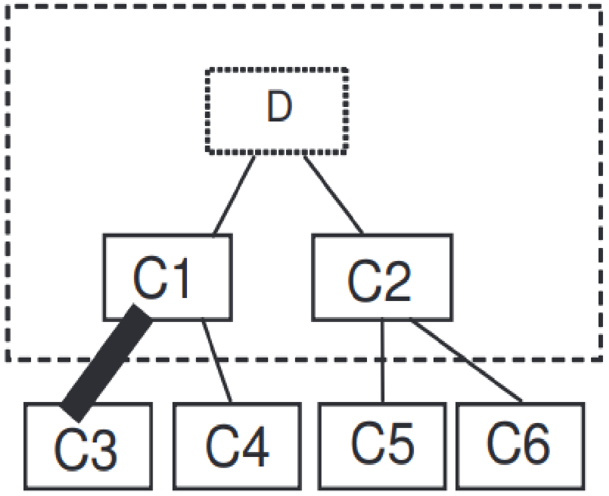
- The entities to integrate may be components as defined in the architectural design or different systems as defined in the product design.
- The principles for integration testing are the same no matter what we are integrating.



- For the collection of interfaces to test an overall integration test plan should be produced specifying, among other things, the order in which this testing is to take place.

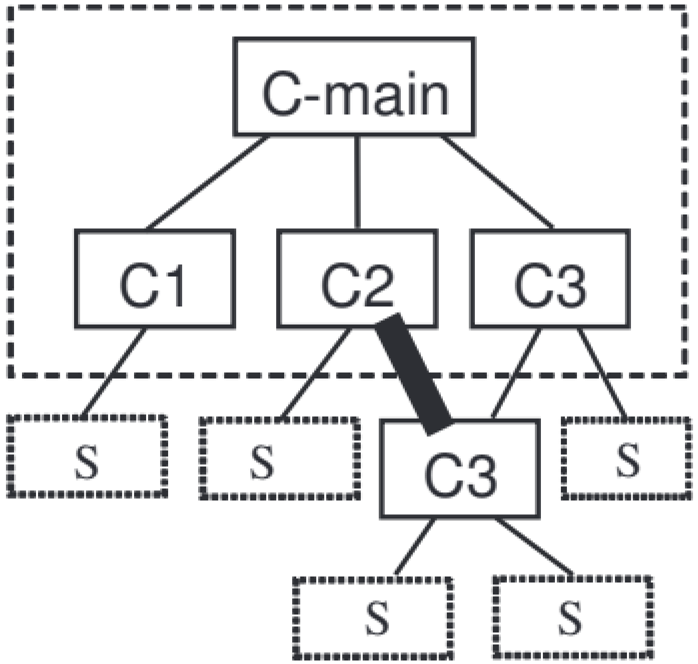
Integration Testing Cont.

- There are four different strategies for the testing order in integration testing:
  - Top down;
  - Bottom up;
  - Functional integration;
  - Big-bang.



- In top-down integration the interfaces in the top layer in the design hierarchy are tested first, followed by each layer going downwards.
- The main program serves as the driver.
- This way we quickly get a "shell" created. The drawback is that we (often) need a large number of stubs.

Integration Testing Cont.



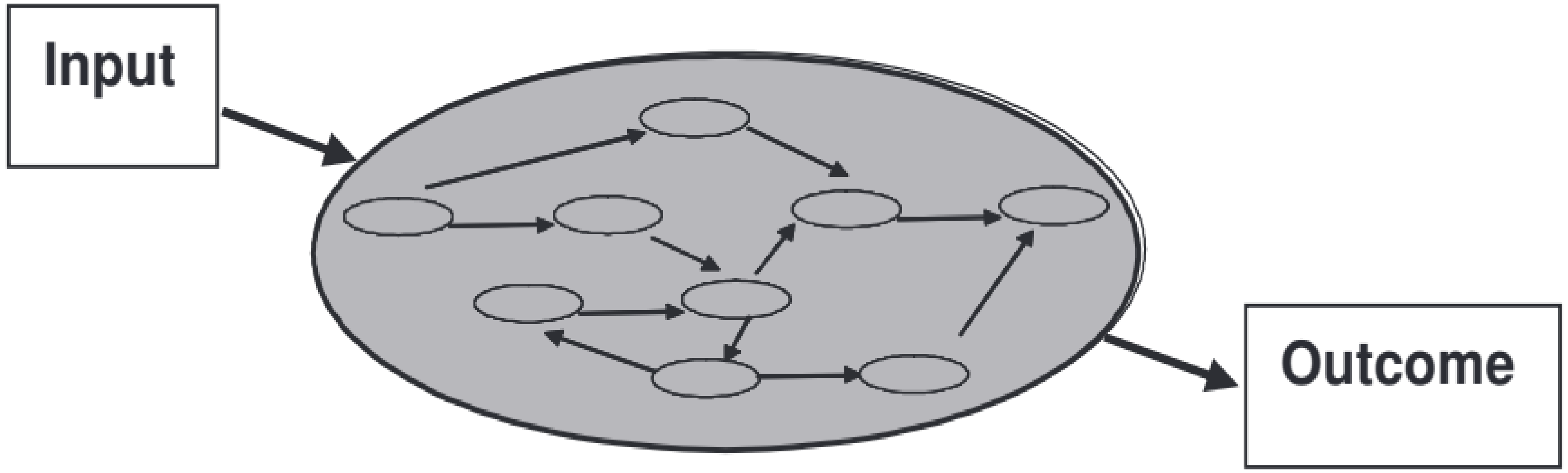
- ➢ In bottom-up integration the interfaces in the lowest level are tested first.
- ➢ Here higher components are replaced with drivers, so we may need many drivers.
- ➢ This integration strategy enables early
- ➢ integration with hardware, where this is relevant.

Integration Testing Cont.

- In functional integration we integrate by functionality area; this is a sort of vertically divided top-down strategy.
- We quickly get the possibility of having functional areas available.
- In big-bang integration we integrate most or everything in one go.
- At first glance it seems like this strategy reduces the test effort, but it does not on the contrary.
- It is impossible to get proper coverage when testing the interfaces in a big-bang integration, and it is very difficult to find any defects in the interfaces, like looking for a needle in a haystack.
- Both top-down and bottom-up integration often end up as big-bang, even if this was not the initial intention.

System Testing

- The goal of system testing is to find defects in features of the system compared to the way it has been defined in the software system requirements.
- The test object is the fully integrated system.



System Testing Cont.

- The better the component testing and the component integration testing has been performed prior to the system testing, the more effective is the system testing.
- All too often system testing is impeded by poor or missing component and component integration testing.
- The system test specification is based on the system requirements specification.
- This is where all the expectations,
  - the functional and
  - the nonfunctional should be expressed.
- The execution of system test follows the completion of the entire component integration testing.

System Testing Cont.

- It is a good idea to also require that a static test has been performed on the requirements specification and on the system test specification before execution starts
- Many tools support system testing.
- Capture/replay tools and test management tools are especially useful to support the system testing.
- Measures of time spent on the testing, on faults found and corrected, and on coverage should be collected.
- The system testing must stop when the completion criteria specified in the plan have been met.
- A system test report should be produced when the system testing has been completed.

Acceptance Testing

- The goal of this test level is not, like for all the other ones, to find defects by getting the product to fail.
- At the acceptance test level the product is expected to be working and it is presented for acceptance.
- The customer and/or end users must be involved in the acceptance testing.
- In the acceptance testing the test object is the entire product. That could include:

Acceptance Testing Cont.

- The techniques are usually mostly experience-based, where the future users apply their domain knowledge and (hopefully) testing skills to the validation of the product.
- There may be a number of acceptance test types, namely:
  - Contract acceptance test;
  - Alpha test;
  - Beta test.

Acceptance Testing Cont.

- The contract acceptance test may also be called factory acceptance test.
- This test must be completed before the product may leave the supplier; the product has to be accepted by the customer.
- It requires that clear acceptance criteria have been defined in the contract.
- An alpha test is usage of the product by representative users at the development site, but reflecting what the real usage will be like.
- Developers must not be present, but extended support must be provided.
- The alpha test is not used particularly often since it can be very expensive to establish a “real” environment.

Acceptance Testing Cont.

- A beta test is usage of the product by selected (or voluntary) customers at the customer site.
- The product is used as it will be in production.
- The actual conditions determine the contents of the test.
- Beta tests preferably run over a longer period of time.
- Beta tests are much used for off-the-shelf products
- the customers get the product early (and possibly cheaper) in return for accepting a certain amount of immaturity and the responsibility for reporting all incidents.

Thank You !!!