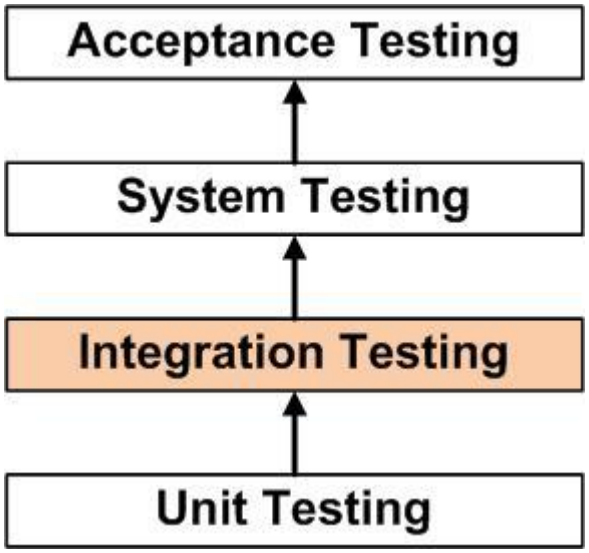
Software Testing

#03 - Testing on Software Life Cycle part-2

Herman Kabetta

Levels of Software Testing



Testing Final System

Testing the Integrated System

Testing Component Groups

Testing Individual Components

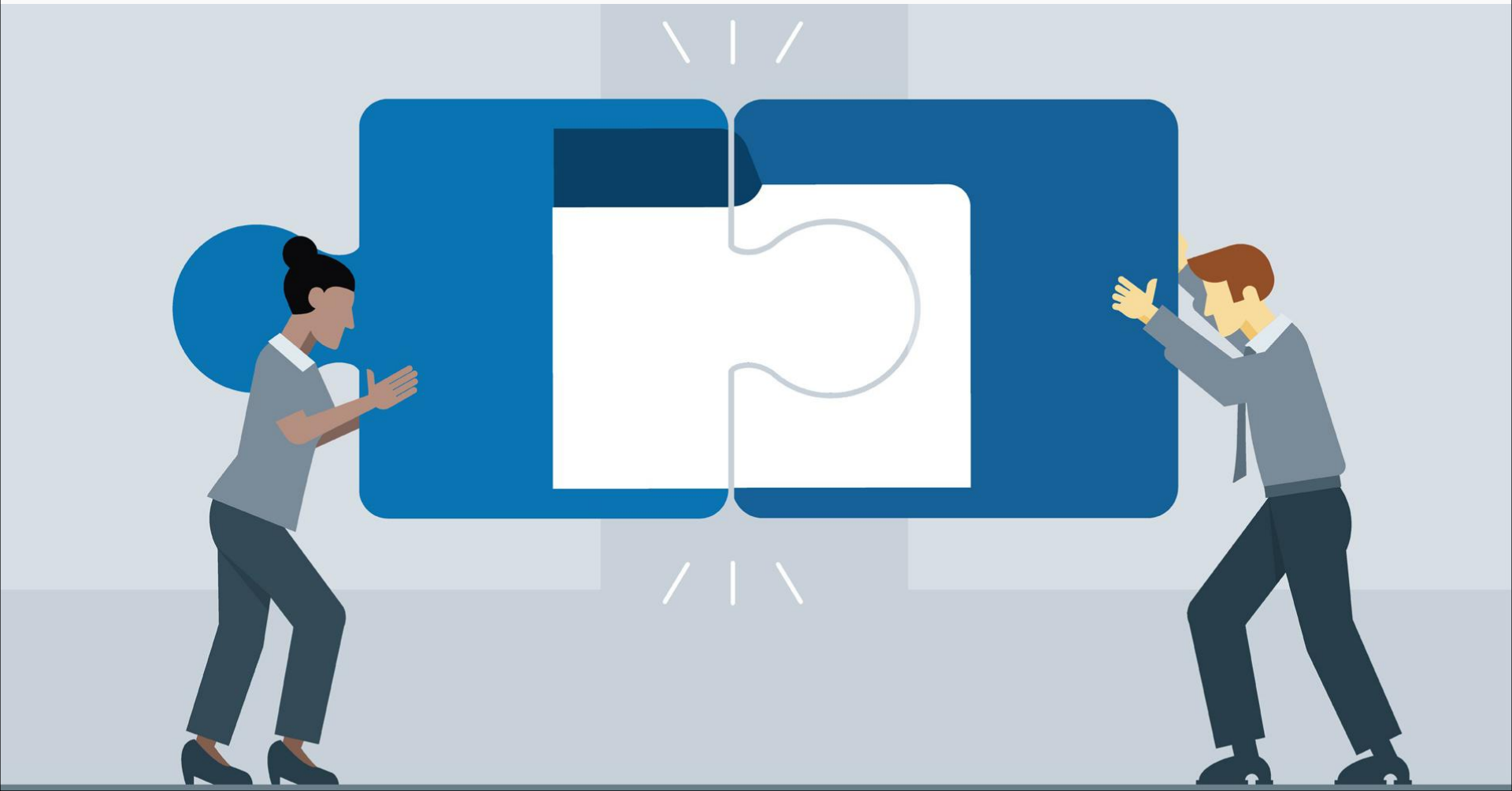
Integration Testing

Although each software module is unit tested, defects still exist for various reasons like

- A Module, in general, is designed by an individual software developer whose understanding and programming logic may differ from other programmers. Integration Testing becomes necessary to verify the software modules work in unity
- At the time of module development, there are wide chances of change in requirements by the clients. These new requirements may not be unit tested and hence system integration Testing becomes necessary.
- Interfaces of the software modules with the database could be erroneous
- External Hardware interfaces, if any, could be erroneous
- Inadequate exception handling could cause issues.

What is Integration Testing?

Integration Testing is a level of software testing where individual units are combined and the connectivity or data transfer between these units is tested. The main aim of this testing is to recognize the interface between the modules.

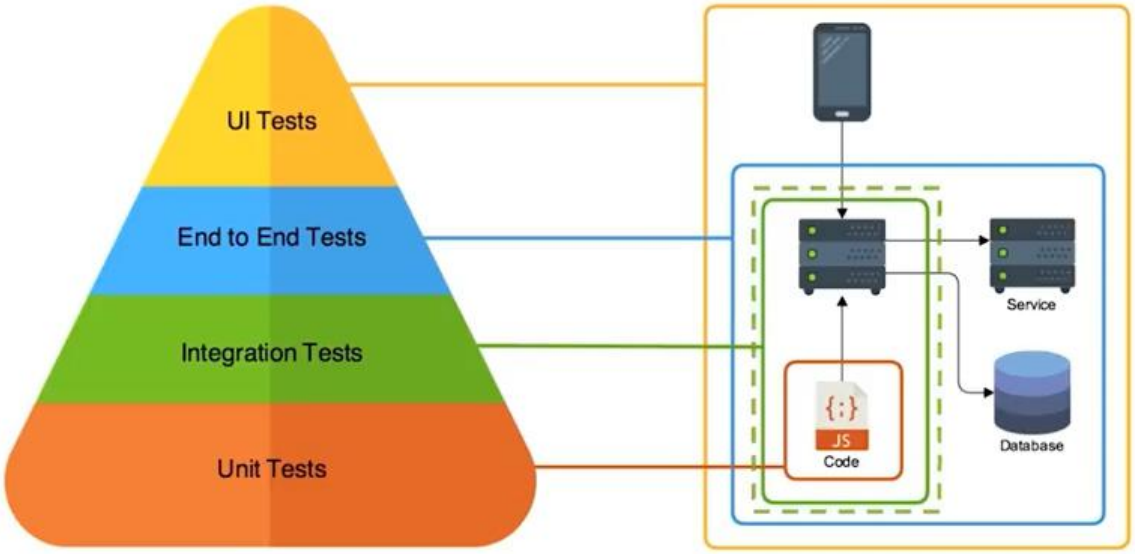


Analogy

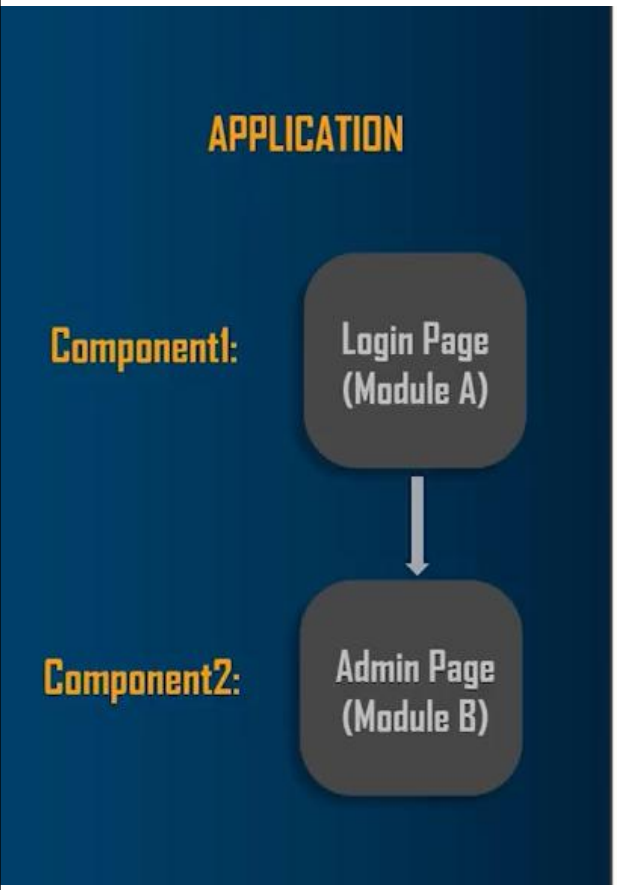
The ink cartridge and the ballpoint are produced separately and unit tested separately. When two or more units are ready, they are assembled and Integration Testing is performed. For example, whether the cap fits into the body or not.

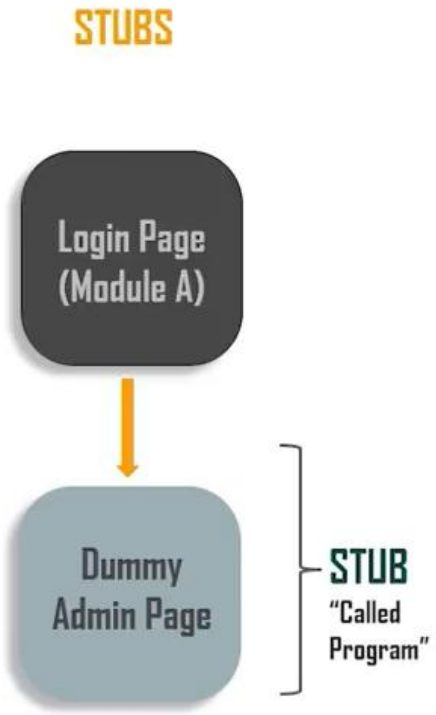
Objectives

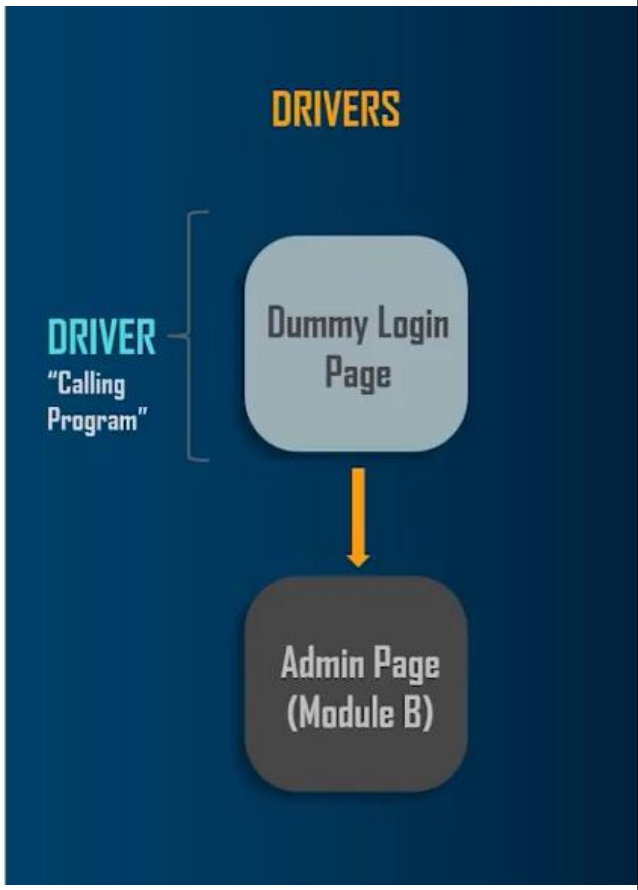
- Ensure modules work together properly
- Helps to uncover errors that lie in interfaces
- Ensure that newly added components are not affected



Different Approuch to Integration Testing







Integration Testing Approaches

Top–Down Approach(TDA)

Testing takes place from top to bottom, following the control flow or architectural structure

Bottom-UP Approach(BUA)

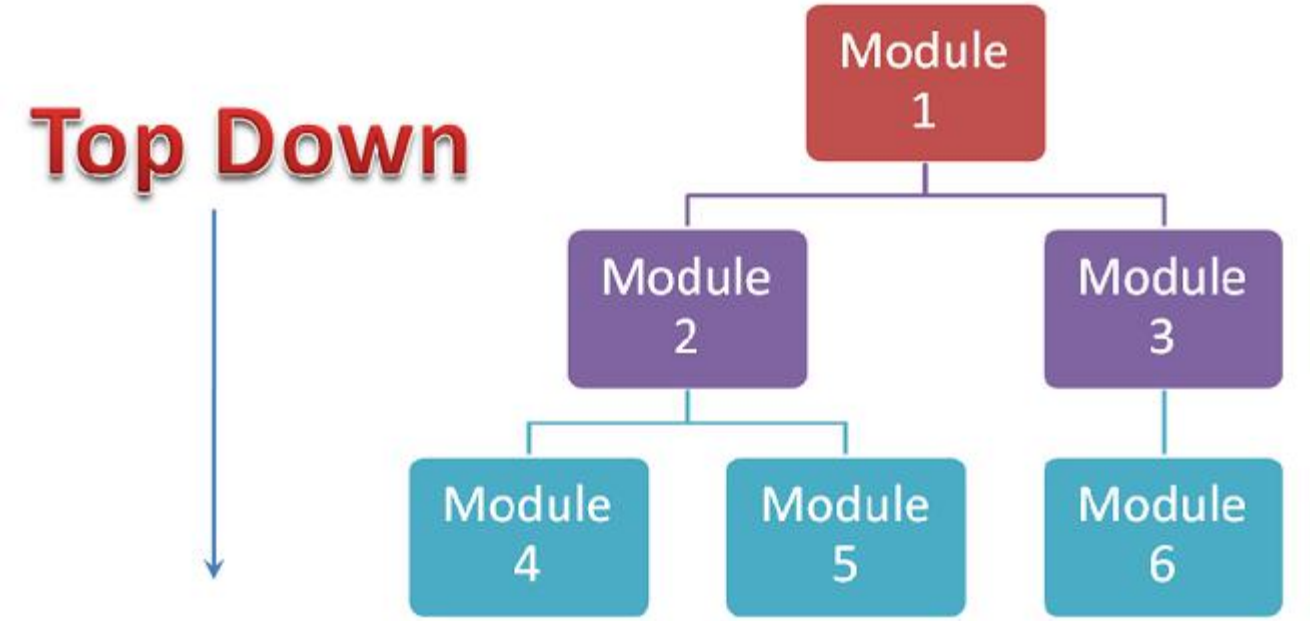
Testing takes place from the bottom of the control flow upwards

Big Bang Approach

All components or modules are integrated simultaneously, after which everything is tested as a whole

Top-Down Approach

In Top to down approach, testing takes place from top to down following the control flow of the software system.



Top-Down Approach

Advantages:

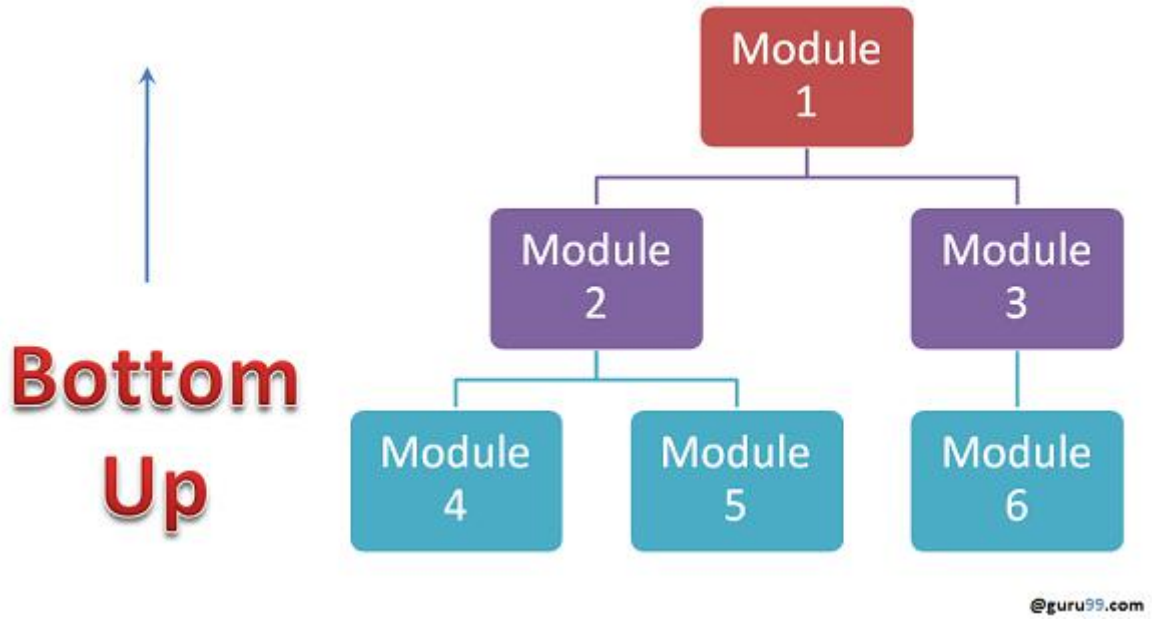
- Fault Localization is easier.
- Possibility to obtain an early prototype.
- Critical Modules are tested on priority; major design flaws could be found and fixed first.

Disadvantages:

- Needs many Stubs.
- Modules at a lower level are tested inadequately.

Bottom-Up Approach

Each module at lower levels is tested with higher modules until all modules are tested. It takes help of Drivers for testing.



Bottom-Up Approach

Advantages:

- Fault localization is easier.
- No time is wasted waiting for all modules to be developed unlike Big-bang approach

Disadvantages:

- Critical modules (at the top level of software architecture) which control the flow of application are tested last and may be prone to defects.
- An early prototype is not possible

Big Bang Approach

All component are integrated together at once and then tested.

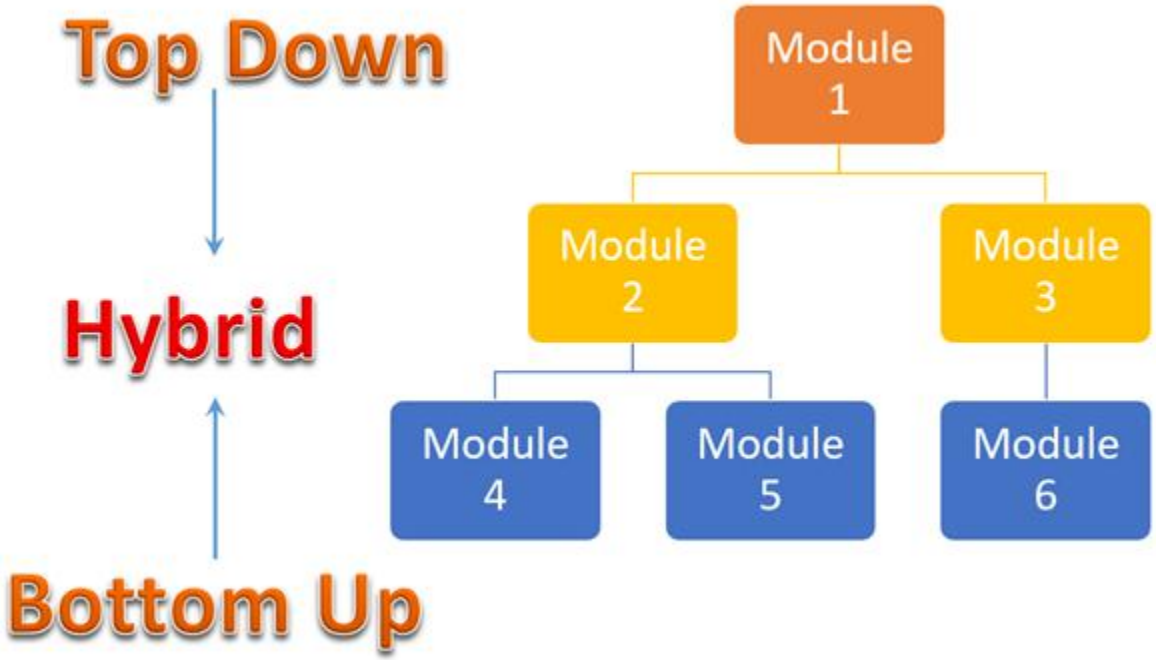
Advantages:

- Convenient for small systems.

Disadvantages:

- Fault Localization is difficult.
- Given the sheer number of interfaces that need to be tested in this approach, some interfaces link to be tested could be missed easily.
- Since the Integration testing can commence only after "all" the modules are designed, the testing team will have less time for execution in the testing phase.
- Since all modules are tested at once, high-risk critical modules are not isolated and tested on priority. Peripheral modules which deal with user interfaces are also not isolated and tested on priority.

Sandwich Integration Approach



Advantage

- Both layers can be tested in parallel

Disadvantage

- High Cost, big skill set, extensive testing is not done

Two Types of Integration Testing

1. Component Integration Testing
   - Focused on the interactions and interfaces between integrated components
   - Performed after unit testing and generally automated
   - Component Integration tests are usually part of the continuous integration process
2. System Integration Testing
   - Int this case, the developing organization does not control the external interfaces, which can create various challenges for testing.
   - System integration testing may be done after system testing or in parallel with ongoing system test activities.

Test Basis of Integration Testing

- Software and system design
- Sequence diagram
- Interface and communication protocol specifications
- Use cases
- Architecture at component or system level
- Workflows
- External interface definitions

Test Objects of Integration Testing

- Subsystem
- Databases
- Infrastructures
- Interfaces
- APIs

Test Defects & Failures of Component Integration Testing

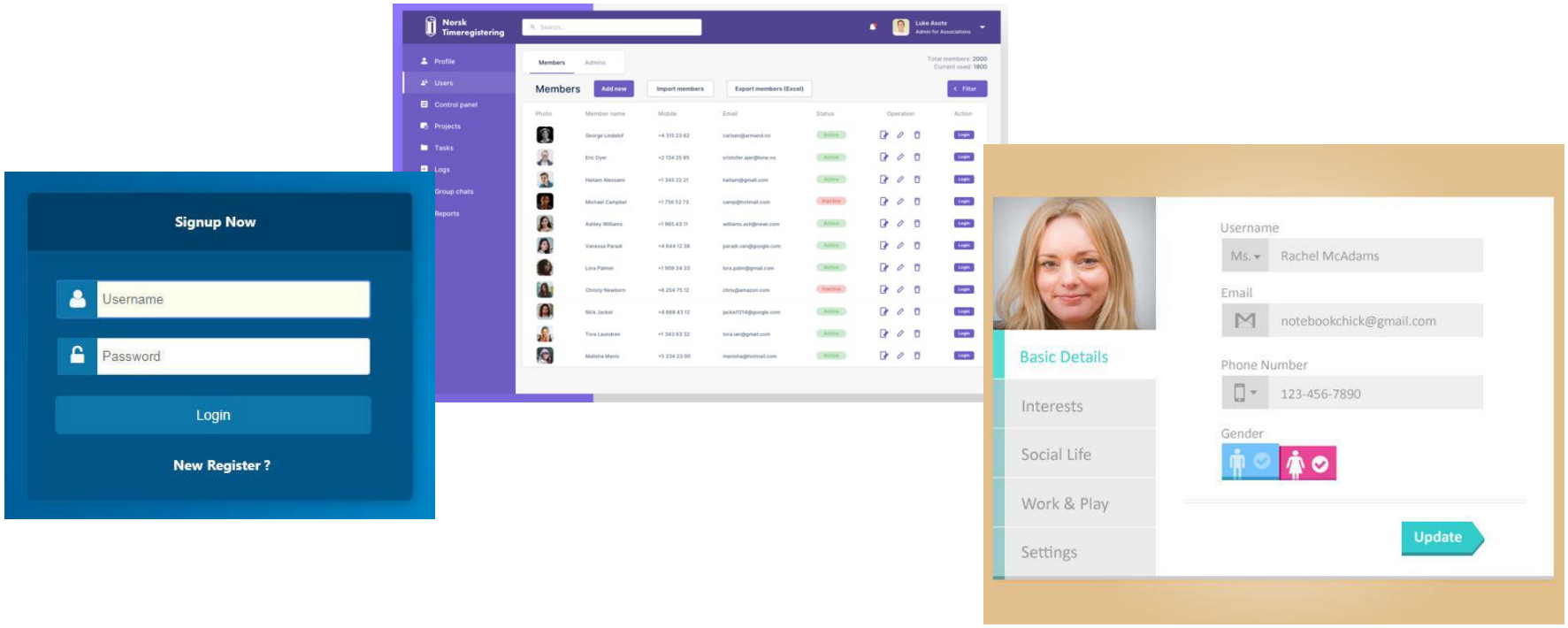
- Incorrect data, missing data or incorrect data encoding
- Incorrect sequencing or timing of interface calls
- Interface mismatch
- Failures in communication betwen components
- Unhandled or improperly handled communication failures between components
- Incorrect assumptions about the meaning,s, or boundaries of the data being passed between components

Test Defects & Failures of System Integration Testing

- Inconsistent message structures between systems
- Incorrect data, missing data or incorrect data encoding
- Interface mismatch
- Failures in communication between systems
- Unhandled or improperly handled communication failures between systems
- Incorrect assumptions about the meaning, units, or boundaries of the data being passed between systems
- Failure to comply with mandatory security regulations

Example Integration Testing

Consider an application with four modules, Login Page, User List, Edit Users and Update Users. All these modules are integrated logically by programmers.



Example Integration Testing

Integration Test Case differs from other test cases in the sense, **it focuses mainly on the interfaces & flow of data/information between the modules**. Here priority is to be given for the integrating links rather than the unit functions which are already tested.

Do not concentrate much on the Login Page testing as it's already been done in Unit Testing. But check how it's linked to the User List Page.

Example Integration Testing

Test Case ID	Test Case Objective	Test Case Description	Expected Result	Actual Result
1	Check the interface link between the Login and User List Page	Enter login credentials and click on the Login button	To be directed to the User List Page	As Expected
2	Check the interface link between the User List Page and Edit User Page	From User List select the user and click on edit button	Detail user should be appear in the Edit Page	As Expected
3	Check the interface link between the Edit User Page and Update User module	Change User details in Edit User Page and click update button	To be directed to the User List Page and show success or failed message	As Expected

Integration Testing Exercise

1. Download any open source project in Github.
2. Do an Integration Testing for the project, below is step by step you must gain :
   a. Draw a control flow diagram of any modules
   b. List all posible Test Cases in a table with expected and actual result
3. e.g. : https://github.com/daniszaidan/InventoryLab

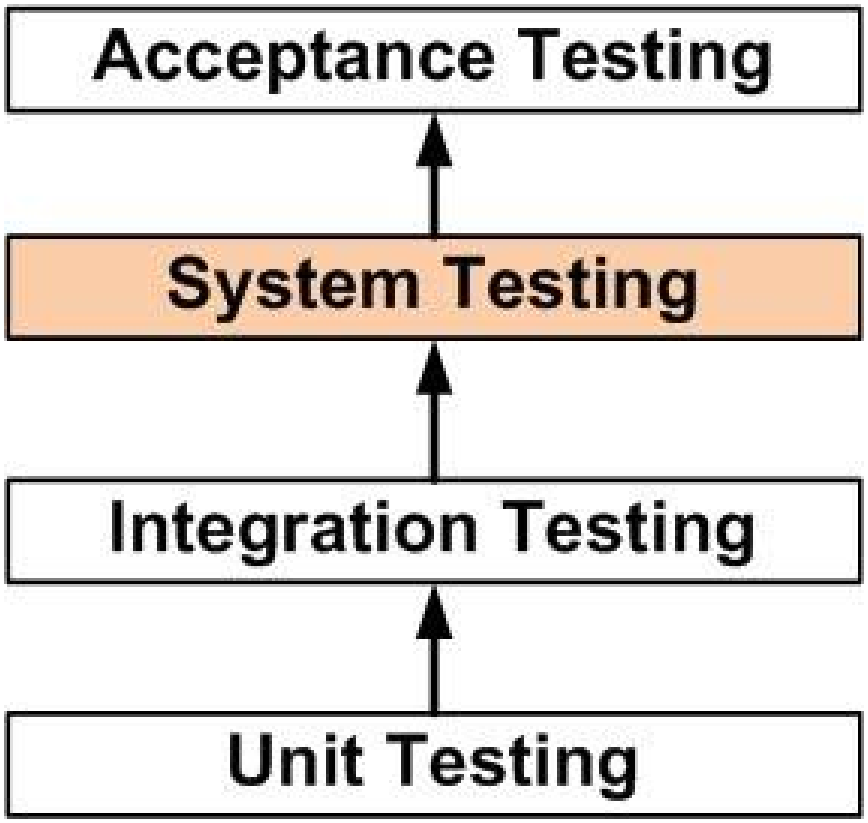
System Testing

Testing of a complete and fully integrated software product.

System Testing focused on the behaviour and capabilities of a whole system or product

Usually, Black Box Testing method is used.

Testing done by a professional testing agent on the completed software product before it is introduced to the market.



System Testing

System Testing involves testing the software code for following

- Testing the fully integrated applications including external peripherals in order to check how components interact with one another and with the system as a whole. This is also called End to End testing scenario.
- Verify through testing of every input in the application to check for desired outputs.
- Testing of the user's experience with the application.

Objectives of System Testing

- Reducing risk
- Verifying wheter the functional and non-functional behaviours of the system are as designed and specified
- Validating that the system is complete and will work as expected
- Building confidence in the quality of the system as a whole
- Finding defects
- Preventing defects from escaping to higher test levels or production

Test Basis of System Testing

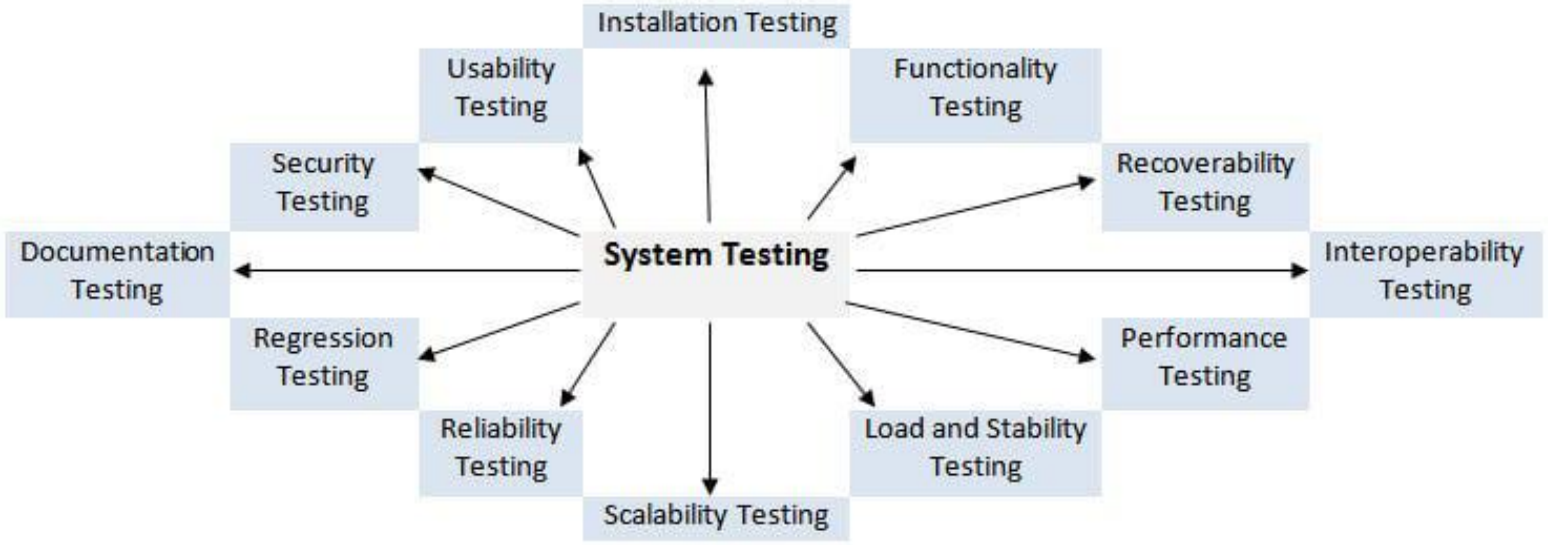
- System and software requirement specifications (functional and non-functional)
- Risk analysis reports
- Use cases
- Epics and user stories
- Models of system behaviour
- State diagrams
- System and user manuals

Test Objects of System Testing

- Applications
- Hardware/Software systems
- Operating systems
- System under test (SUT)
- System configuration and configuration data

Different Types of System Testing

There are more than 50 types of System Testing.



What Should Testers Use?

The specific types used by a tester depend on several variables. Those variables include:

- **Who the tester works for** – This is a major factor in determining the types of system testing a tester will use. Methods used by large companies are different than that used by medium and small companies.
- **Time available for testing** – Ultimately, all 50 testing types could be used. Time is often what limits us to using only the types that are most relevant for the software project.
- **Resources available to the tester** – Of course some testers will not have the necessary resources to conduct a testing type. For example, if you are a tester working for a large software development firm, you are likely to have expensive automated testing software not available to others.

What Should Testers Use?

The specific types used by a tester depend on several variables. Those variables include:

- **Software Tester's Education** - There is a certain learning curve for each type of software testing available. To use some of the software involved, a tester has to learn how to use it.
- **Testing Budget** - Money becomes a factor not just for smaller companies and individual software developers but large companies as well.

Discussion

1. Usability testing
2. Performance testing
3. Regression testing
4. Security testing
5. Sanity testing
6. Smoke testing

Including example and test plan template for each types of testing

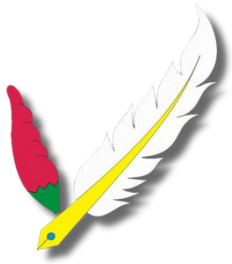
Functionality Testing

- Functional Testing is defined as a type of testing which verifies that each function of the software application operates in conformance with the requirement specification.
- Mainly involves black box testing and it is not concerned about the source code of the application.

Functionality Testing Objectives

- **Mainline functions:** Testing the main functions of an application
- **Basic Usability:** It involves basic usability testing of the system. It checks whether a user can freely navigate through the screens without any difficulties.
- **Accessibility:** Checks the accessibility of the system for the user
- **Error Conditions:** Usage of testing techniques to check for error conditions. It checks whether suitable error messages are displayed.

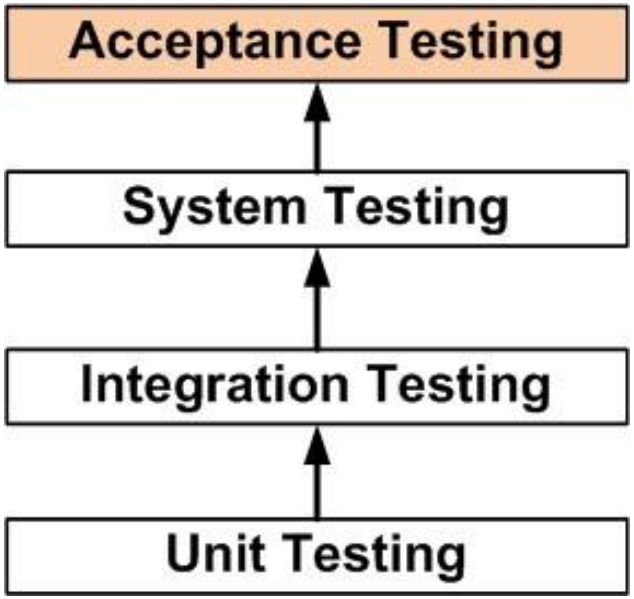
	Unit testing	Integration testing	Functional testing
Definition and purpose	Testing smallest units or modules individually.	Testing integration of two or more units/modules combined for performing tasks.	Testing the behavior of the application as per the requirement.
Complexity	Not at all complex as it includes the smallest codes.	Slightly more complex than unit tests.	More complex compared to unit and integration tests.
Testing techniques	White box testing technique.	White box and black box testing technique. Grey box testing	Black box testing technique.
Major attention	Individual modules or units.	Integration of modules or units.	Entire application functionality.
Error/Issues covered	Unit tests find issues that can occur frequently in modules.	Integration tests find issues that can occur while integrating different modules.	Functional tests find issues that do not allow an application to perform its functionality. This includes some scenario-based issues too.
Issue escape	No chance of issue escape.	Less chance of issue escape.	More chances of issue escape as the list of tests to run is always infinite.



Functional Testing	Non-Functional Testing
Functional testing is performed using the functional specification provided by the client and verifies the system against the functional requirements.	Non-Functional testing checks the Performance, reliability, scalability and other non-functional aspects of the software system.
Functional testing is executed first	Non-functional testing should be performed after functional testing
Manual Testing or automation tools can be used for functional testing	Using tools will be effective for this testing
Business requirements are the inputs to functional testing	Performance parameters like speed, scalability are inputs to non-functional testing.
Functional testing describes what the product does	Nonfunctional testing describes how good the product works
Easy to do Manual Testing	Tough to do Manual Testing

Acceptance Testing

- Another last test levels
- Focus on the cutomer's and user's perspective
- The purpose of this test is to evaluate the system’s compliance with the business requirements and assess whether it is acceptable for delivery



Acceptance Testing

- **Internal Acceptance Testing** (Also known as Alpha Testing) is performed by members of the organization that developed the software but who are not directly involved in the project (Development or Testing). Usually, it is the members of Product Management, Sales and/or Customer Support.
- **External Acceptance Testing** is performed by people who are not employees of the organization that developed the software.
  - **Customer Acceptance Testing** is performed by the customers of the organization that developed the software. They are the ones who asked the organization to develop the software. [This is in the case of the software not being owned by the organization that developed it.]
  - **User Acceptance Testing** (Also known as Beta Testing) is performed by the end users of the software. They can be the customers themselves or the customers' customers.

Basis of Acceptance Testing

- User or system requirements
- Use case
- Business process
- Risk analyses
- User process description
- Laws and regulations

Acceptance Testing

Case 1: User Account handling

This is the scenario where the users are allowed to Create, View, Update, and De-activate their account. In general, it's a CRUD operation (Create, Read, Update, and Delete). So directly we will get 4 major scenarios to test.

Along with this, in real-time user account handling, we have many areas when it comes to viewing and updating.

Proceeding with writing acceptance tests:

Test 1: Registration/Sign Up/Create Account, verify if a User is able to:

- Create the Account.
- Activate the account.
- Activate the account only once (Here, activation link has to be tested for 2nd Even though this is negative testing, it is one of the major verification points to be considered).

Test 2: To Access and View Account information, verify if a User is able to:

- Log in to the account.
- View different sections in the Profile (If Profile section is categorized, then each and every category should be viewable).
- Verify that the data displayed in Profile is correct as per the User's input.

Acceptance Testing

Case 2: Purchasing Product

Purchasing of the product usually has the general flow.

Some general scenarios what end-users look at are listed here:

Pre-condition: User should be logged in to the application.

Test 1: Product Details, verify if a User is able to:

- View the Product details page.
- View all the sub-sections in the Product details page (Description, Feature, Brand information, etc.).
- Select the Quantity of the product, Color, Size, etc. as available in the Product details page.
- Navigate to the category, sub-category pages from the Product Details page (if available in Product details page).
- Navigate to the other Product's details page (if provided relevant products section).
- View comments and ratings on the product.
- Sort Comments of the Product based on ratings.
- View overall rating of the Product.
- Add Comment on the Product.
- Update his/her comment on the product.
- Delete his/her comment on the product (if provided).

Acceptance Testing

Test 2: Add to Cart, verify if a User is:

- Able to add the product to Cart:
  - Through Product details page.
  - Through Product list page.
- Able to add required quantity to the cart (1 to max limit set).
- Not able to add the product to the Cart if Out-of-Stock.

Test 3: In the Cart Page, verify if a User is able to:

- View the Product in the Cart with Price details for added quantity.
- Update quantity (1 to max limit set).
- Remove the Product from Cart.
- Navigate back to shopping.
- Continue to Checkout.
- View Empty Cart when no product is added,

Reviewing Acceptance Testing

Reviewing Acceptance tests is an important task as it needs to be correct and to-the-point with respect to the business requirements. As these may be conducted by Customers themselves and/or end-users, it is very much necessary to be complete, non-ambiguous, correct, and detailed enough for anyone to understand and execute.

Reviewing Acceptance tests has to be done by Business Analysts, Customers and any review comments should be incorporated into high priority.

Functionality Test Plan

Test Case ID	FT_001	**Specification**	Test the Login Functionality		
Created By	Ignatius	**Reviewed By**	Yodha		
Tester's Name	Ignatius	**Date Tested**	1-Jan-2019	**Test Case (Pass/Fail/Not**	Pass

S #	Prerequisites:
1	Access to Chrome Browser
2	System is already using https
3	
4	

S #	Test Data
1	Userid = mg12345
2	Pass = df12@434c
3	
4	

Test Scenario Verify on entering valid userid and password, the customer can login

Step #	Step Details	Expected Results	Actual Results	Pass / Fail / Not Run / Blocked
1	Navigate to http://demo.example.com	Site should open	As Expected	Pass
2	Enter Userid & Password	Credential can be entered	As Expected	Pass
3	Click Submit	Cutomer is logged in	As Expected	Pass