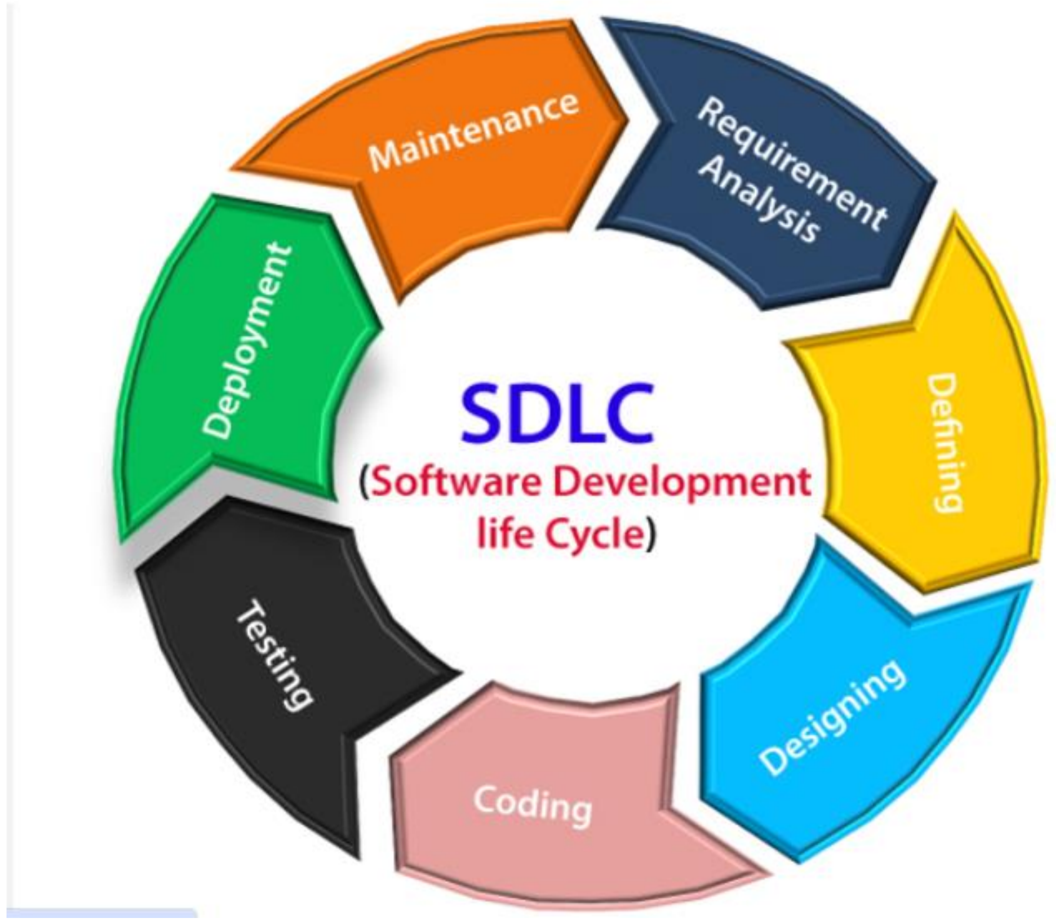
Software Testing

Introduction

Software Testing is a procedure to verify whether the actual results as same as of expected results.

Software testing is performed to provide assurance that the software system does not contain any defects.

The engineer creates a series of test cases that are intended to "defeat" the software that has been built.



Testing Objectives

1. Testing is a process of executing a program with the intent of finding an error.

2. A good test case is one that has a high probability of finding an as-yet undiscovered error.

3. A successful test is one that uncovers an as-yet-undiscovered error.

Advantages of Software Testing

- ❖Reduce the possibility of software failure
- ❖Remove maximum possible errors
- ❖Correctness and completeness of software
- ❖Verify and validate software
- ❖Produce expected results

Testing Principles

- ❖All tests should be traceable to customer requirements.
- ❖Tests should be planned long before testing begins.
- ❖The Pareto principle applies to software testing.
- ❖Testing should begin "in the small" and progress toward testing "in the large."
- ❖Exhaustive testing is not possible.
- ❖To be most effective, testing should be conducted by an independent third party.

Testing Tactics

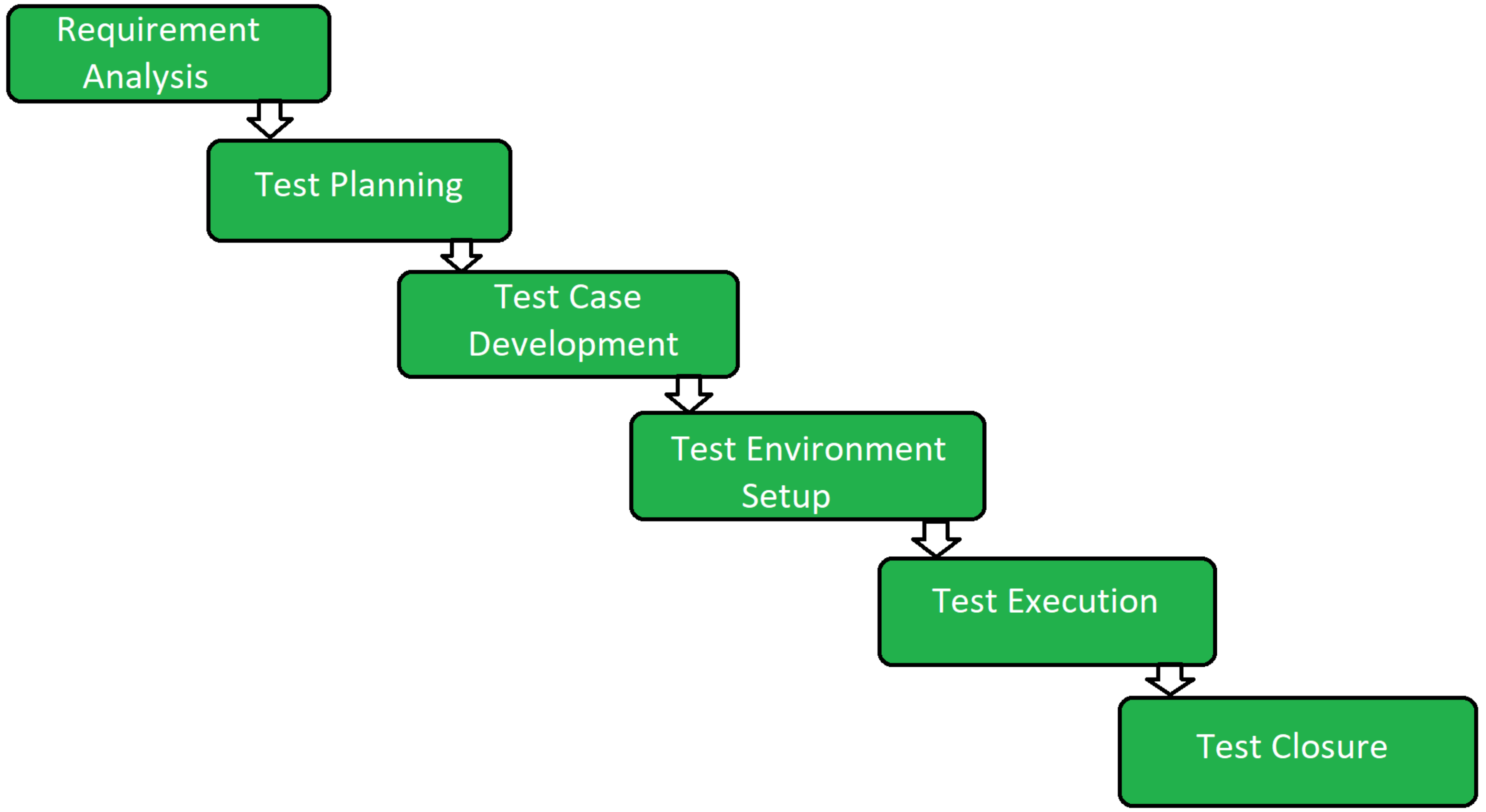
Types of Testing

1. manual Testing

2. Automation testing

Manual Testing	Automated Testing
Manual testing requires human intervention for test execution.	Automation Testing is use of tools to execute test cases
Manual testing will require skilled labour, long time & will imply high costs.	Automation Testing saves time, cost and manpower. Once recorded, it’s easier to run an automated test suite
Any type of application can be tested manually, certain testing types like ad-hoc and monkey testing are more suited for manual execution.	Automated testing is recommended only for stable systems and is mostly used for Regression Testing
Manual testing can become repetitive and boring.	The boring part of executing same test cases time and again is handled by automation software in Automation Testing.

Software Testing Process



Software Testing Life Cycle (STLC) is a **sequence of specific activities** conducted during the testing process to ensure software quality goals are met.

STLC involves both **verification and validation** activities.

Software Testing is **not just a single/isolate** activity, i.e. testing.

It consists of a **series of activities** carried out methodologically to help certify your software product.

Each of these stages has a definite Entry and Exit criteria, Activities & Deliverables associated with it.

Entry Criteria: Entry Criteria gives the prerequisite items that must be completed before testing can begin.

Exit Criteria: Exit Criteria defines the items that must be completed before testing can be concluded

Requirement Analysis

Requirement Analysis in which test team studies the requirements from a testing point of view to identify testable requirements and the QA team may interact with various stakeholders to understand requirements in detail. Requirements could be either functional or non-functional.

Activities in Requirement Phase Testing

1. Gather details about testing priorities and focus.
2. Identify types of tests to be performed.
3. Prepare **Requirement Traceability Matrix (RTM).**
4. Identify test environment details where testing is supposed to be carried out.

Deliverables of Requirement Phase Testing

1. RTM
2. Automation feasibility report. (if applicable)

Test Planning

Test Planning in STLC is a phase in which a Senior QA manager determines the test plan strategy along with efforts and cost estimates for the project. The resources, test environment, test limitations and the testing schedule are also determined. The Test Plan gets prepared and finalized in the same phase.

Test Planning Activities

1. Preparation of test plan/strategy document for various types of testing
2. Test tool selection
3. Test effort estimation
4. Resource planning and determining roles and responsibilities.
5. Training requirement

Deliverables of Test Planning

1. Test plan/strategy document.
2. Effort estimation document.

Test Case Development Phase

- The **Test Case Development Phase** involves the creation, verification and rework of test cases & test scripts after the test plan is ready.
- Initially, the Test data is identified then created and reviewed and then reworked based on the preconditions.
- Then the QA team starts the development process of test cases for individual units.

Test Case Development Activities

1. Create test cases, automation scripts (if applicable)
2. Review test cases and scripts
3. Create test data (If Test Environment is available)

Deliverables of Test Case Development

1. Test cases/scripts
2. Test data

Test Environment Setup

Test Environment Setup decides the software and hardware conditions under which a work product is tested.

The test team is required to do a readiness check (smoke testing) of the given environment.

Test Environment Setup Activities

1. Understand the required architecture, environment set-up and prepare hardware and software requirement list for the Test Environment.
2. Setup test Environment and test data
3. Perform smoke test on the build

Deliverables of Test Environment Setup

1. Environment ready with test data set up
2. Smoke Test Results.

Test Execution Phase

Test Execution Phase is carried out by the testers in which testing of the software build is done based on test plans and test cases prepared.

The process consists of test script execution, test script maintenance and bug reporting.

If bugs are reported then it is reverted back to development team for correction and retesting will be performed.

Test Execution Activities

1. Execute tests as per plan
2. Document test results
3. Map defects to test cases in RTM
4. Retest the Defect fixes
5. Track the defects to closure

Deliverables of Test Execution

1. Completed RTM with the execution status
2. Test cases updated with results
3. Defect reports

Test Cycle Closure

Test Cycle Closure phase is completion of test execution which involves several activities like test completion reporting, collection of test completion matrices and test results.

Testing team members meet, discuss and analyze testing artifacts to identify strategies that have to be implemented in future, taking lessons from current test cycle.

Test Cycle Closure Activities

1. Evaluate cycle completion criteria based on Time, Test coverage, Cost, Software, Critical Business Objectives, Quality
2. Prepare test metrics based on the above parameters.
3. Document the learning out of the project
4. Prepare Test closure report

Deliverables of Test Cycle Closure

1. Test Closure report
2. Test metrics

A Strategic Approach to Testing

- ❑A strategy for software testing integrates the design of software test cases into a well-planned series of steps that result in successful development of the software
- ❑The strategy provides a road map that describes the steps to be taken, when, and how much effort, time, and resources will be required
- ❑The strategy incorporates test planning, test case design, test execution, and test result collection and evaluationM

General Characteristics of Strategic Testing

- ❑To perform effective testing, a software team should conduct effective formal technical reviews
- ❑Testing begins at the component level and work outward toward the integration of the entire computer-based system
- ❑Different testing techniques are appropriate at different points in time
- ❑Testing is conducted by the developer of the software and (for large projects) by an independent test group
- ❑Testing and debugging are different activities, but debugging must be accommodated in any testing strategy

Verification and Validation

Software testing is part of a broader group of activities called verification and validation that are involved in software quality assurance

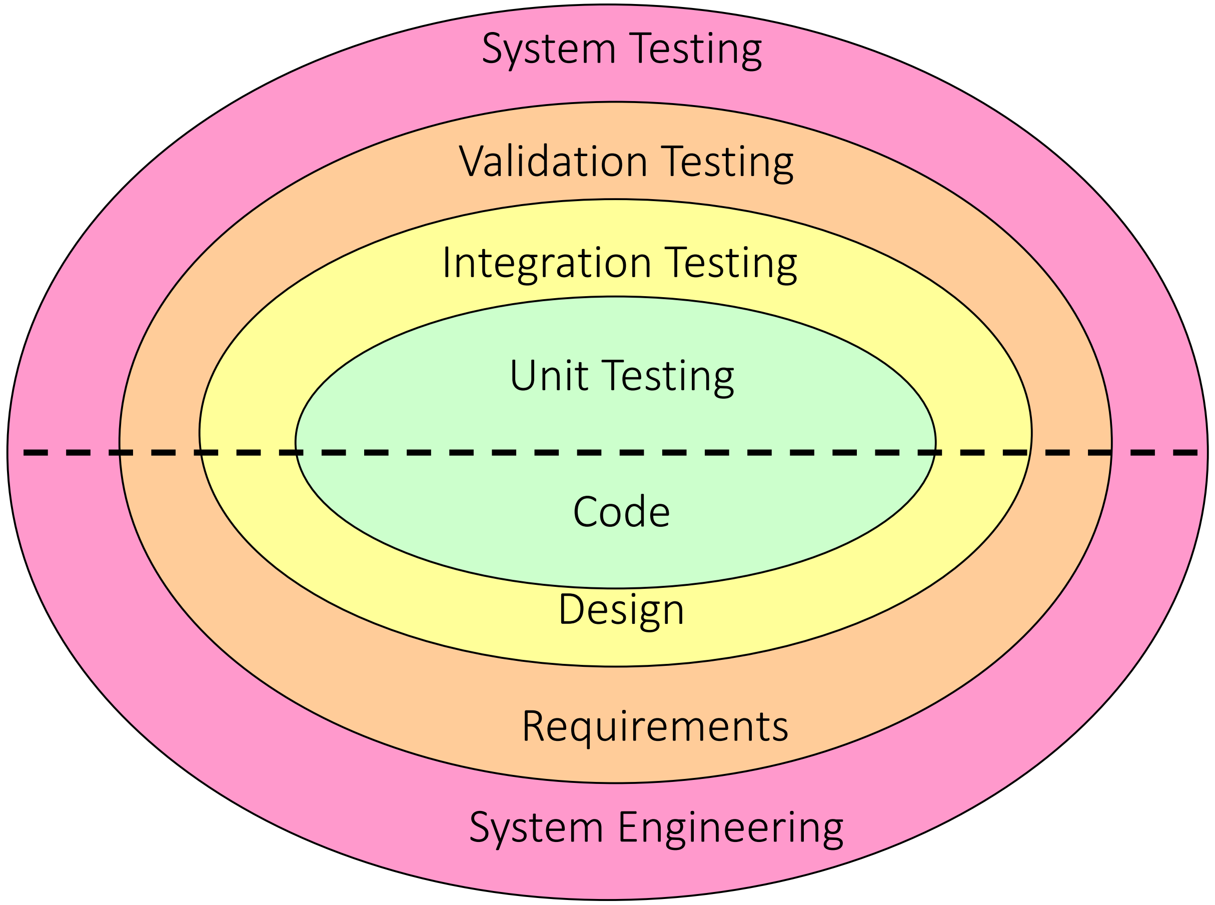
Verification (Are the algorithms coded correctly?)

The set of activities that ensure that software correctly implements a specific function or algorithm

Validation (Does it meet user requirements?)

The set of activities that ensure that the software that has been built is traceable to customer requirements

A Strategy for Testing Conventional Software



Levels of Testing for Conventional Software

Unit testing

Concentrates on each component/function of the software as implemented in the source code

Integration testing

Focuses on the design and construction of the software architecture

Validation testing

Requirements are validated against the constructed software

System testing

The software and other system elements are tested as a whole

Unit Testing

Unit Testing is a type of software testing where individual units or components of a software are tested.

The purpose is to validate that each unit of the software code performs as expected.

Unit Testing is done during the development (coding phase) of an application by the developers.

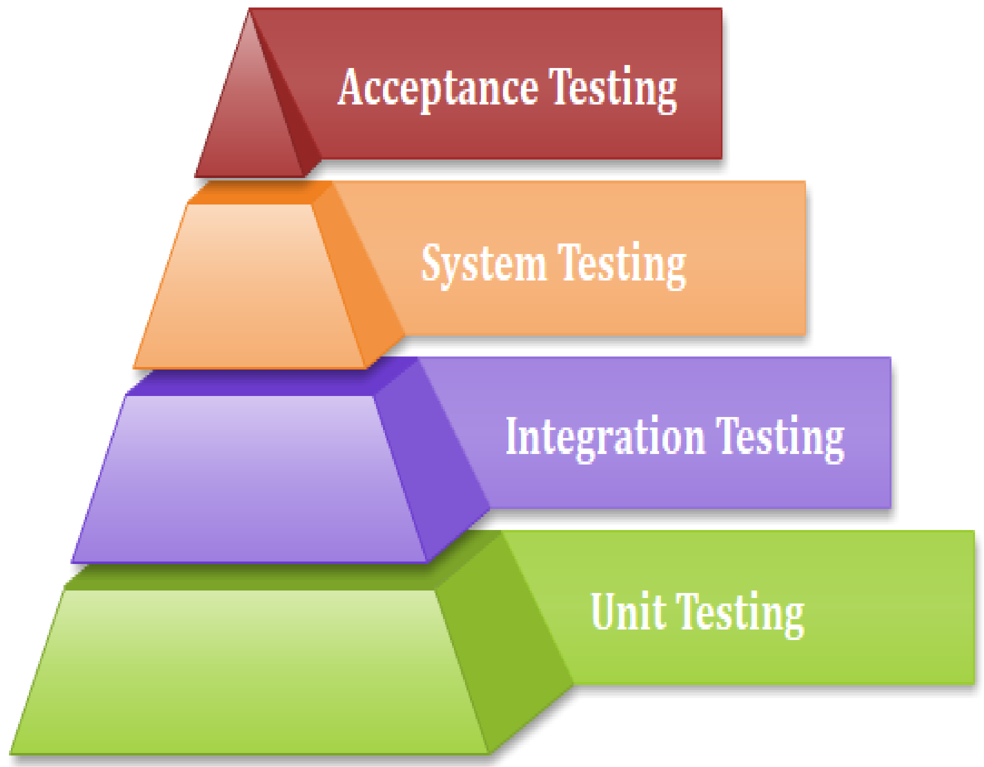
Unit Tests isolate a section of code and verify its correctness.

A unit may be an individual function, method, procedure, module, or object.

Why Unit Testing?

In a testing level hierarchy, unit testing is the first level of testing done before integration and other remaining levels of the testing.

It uses modules for the testing process which reduces the dependency



Unit Testing Method :

It is performed by the White Box testing methods.

When is it performed?

Unit Testing is first level of software testing and is performed prior to integration testing

Who preforms it?

It is normally performed by software developed. It may also be performed by testers

Stubs in Unit Testing

Example:

We have 3 modules – A, B abd C

Module A is ready and we need to test it. But module B and C is not ready.

Module A calls functions from Module B and C.

So developer write a dummy code which simulates B and C.

This dummy code is known as stub

Drivers in Unit Testing

Ex

Module B and C ready but Module A which calls functions from Module B and C is not ready.

Developer will write a dummy code for Module A which return values to Module B and C.

This dummy code is known as driver.