Software Testing and Maintenance

- ## Testing: Software Quality

Importance of testing

To verify whether the actual results are the same as expected results.

To assure that the software system does not contain any defects.

To ensure that all user requirements are fulfilled by software.

To give assurance that we deliver quality products to customers.

Software quality depends on to what extent software fulfills user requirements and the number of defects occurred in software.

Test cases and test data are created to perform software testing. A test case refers to the actions required to verify a specific feature or functionality in software testing.

Collection of test cases is called a test suit.

Test Data is the input given to a software program during test execution. It represents data that affects or is affected by software execution while testing. Test data is used for both positive testing to verify that functions produce expected results for given inputs and for negative testing to test software ability to handle unusual, exceptional or unexpected inputs.

Software testing is one element of a broader topic that is often referred to as verification and validation

Verification

Verification refers to the set of activities that ensure that software correctly implements a specific function. Verification is the process of evaluating the intermediary work products of a software development lifecycle to check if we are on the right track of creating the final product. Verification: "Are we building the product right?"

Validation

Validation refers to a different set of activities that ensure that the software that has been built is traceable to customer requirements. Validation is the process of evaluating the final product to check whether the software meets the business needs. Validation: "Are

we building the right product?"

Verification	Validation
Verification is the process to find whether the software meets the specified requirements for particular phase.	The validation process is checked whether the software meets requirements and expectation of the customer.
It estimates an intermediate product.	It estimates the final product.
The objectives of verification is to check whether software is constructed according to requirement and design specification.	The objectives of the validation is to check whether the specifications are correct and satisfy the business need.
It describes whether the outputs are as per the inputs or not.	It explains whether they are accepted by the user or not.
Verification is done before the validation.	It is done after the verification.
Plans, requirement, specification, code are evaluated during the verifications.	Actual product or software is tested under validation.
It manually checks the files and document.	It is a computer software or developed program-based checking of files and document.

Advantages of Software Testing

- Earlier Defects Detection
- Increased Customer Satisfaction
- Cost Reduction
- Improved Product Quality and Reliability
- Quicker Development Process
- Enhanced Security
- Easier Recovery

Manual Testing

Manual testing is testing of the software where tests are executed manually by a QA Analyst. It is performed to discover bugs in software under development.

In Manual testing, the tester checks all the essential features of the given application or software. In this process, the software testers execute the test cases and generate the test reports without the help of any automation software testing tools.

It is a classical method of all testing types and helps find bugs in software systems. It is generally conducted by an experienced tester to accomplish the software testing process.

First we do manual testing for any new application before going for automation testing.

Manual testing requires more effort.

Goals of Manual Testing

- to assure that our software product does not contain any defect and it fulfills all functional requirements of the end user.

- test suits (collection of test cases) or cases are designed in the testing phase to test all functionalities of the product.

- to assure that reported defects are fixed by the developer and tester performs retesting on it after fixing the defects.

- mainly manual testing verifies the quality of the software product and deploys bug-free software to the customer.

List of manual testing :

- Unit testing
- Integration testing
- System testing
- Acceptance testing

Automation Testing

In Automated Software Testing, testers use automation tools to run our test cases to find out bugs from the software product.

Automated testing entirely relies on the pre-scripted test which runs automatically to compare actual results with the expected results. This helps the tester to determine whether or not an application performs as expected.

Automated testing allows you to execute repetitive tasks and regression tests without the intervention of manual testers. Even though all processes are performed automatically, automation requires some manual effort to create initial testing scripts.

Automation testing requires a considerable amount of money and resources such as

employees, testing tools etc.

We can record test suits by using test automation tools and replay it when needed.

Aim of Automation testing is to decrease the number of test cases to be executed manually but not replace Manual Testing completely.

Automation testing is performed for projects if requirements of projects are stable to some extent i.e. requirements are not frequently changing.

List of some Automation tools

- Selenium
- Mentis
- Quality Test Professional (QTP)
- Bugzilla
- HP ALM (Application Lifecycle Management)

Software Testing Process

Software Testing process activities

- Requirement analysis
- Test Planning
- Test Case development
- Environment set up
- Test execution
- Test cycle closures

Each of the activities of the testing process has defined entry and exit criteria.

Entry criteria

Entry criteria of testing are prerequisite conditions that must be satisfied before testing begins

Entry criteria

An exit criteria of testing describes the conditions that must be satisfied before testing is concluded.

Software testing levels are the different stages of the software development lifecycle where testing is conducted.

There are 4 levels of software testing : Unit -> Integration -> Validation Testing -> System Testing.

Unit Testing

- individual units/ components of a software are tested.

- to validate that each unit of the software performs as designed.

- A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output.

It is the first level of software testing and is performed prior Integration Testing.

It is normally performed by software developers themselves or their peers. In some cases, it may also be performed by independent software testers.

In procedural programming, a unit may be an individual program, function, procedure etc.

In object oriented programming, the smallest unit is a method, which may belong to a base class, abstract class or derived class.

Unit Testing Method

It is performed by using the White Box Testing method.

Unit Test environment

STUBS

Assume we have 3 modules, A, B and C.

Module A is ready and we need to test it, but module A calls functions

from Module B and Module C which is not ready.

So developer will write a dummy module which simulates b and C and

returns values to module A. This dummy module code is known as Stub

DRIVERS

Now suppose we have Modules B and C ready but Module A which calls functions from Module B and Module C is not ready so the developer will write a dummy piece code for Module A which will return values to Module B and C. This dummy piece of code is known as Driver.

Logically both driver and stubs are the software that are written but not submitted to the customer and thus are considered as the overhead. So it is recommended to keep these overhead simple to reduce the cost.

Integration Testing

- a method of software testing in which all units of software under test are integrated and tested as a single group.

- always done after unit testing

- focuses on testing data communication among all units of the system.

- main objective is to determine faults in communication between integrated units.

- make use of methodologies like Big bang approach , incremental approach (top-down, bottom-up or combination of both).

Types of Integration Testing

1. Big bang approach

In this method, all modules of a software product are created first and then they are combined together and the whole software is tested at once.

Advantage

Suitable for small software projects.

Disadvantages

Finding defects is a difficult task because we test the whole software at once.

There are lots of interfaces which need to be tested.

There is possibility that some interfaces links may remain untested

Critical modules are not separated and tested on priority basis because all modules are tested at the same time.

Testing team gets a small amount of time because in this approach, integration testing starts after all the modules are developed.

2. Incremental Approach

In this method, to perform testing two or more modules are merged with each other which are logically related.

Then other associated modules are included in this group of modules and perform testing for checking whether they are functioning correctly or not.

This procedure continues until all modules are grouped together and tested successfully.

This procedure is done with the help of dummy programs which are known as Stubs and Drivers.

Stubs and Drivers do not contain the complete programming logic of the module but they contain code which is needed to perform communication with other modules.

Stub is a dummy program which is called by the Module on which testing is performed.

Driver is a dummy program which calls another module.

Incremental approach is performed using two different methods:

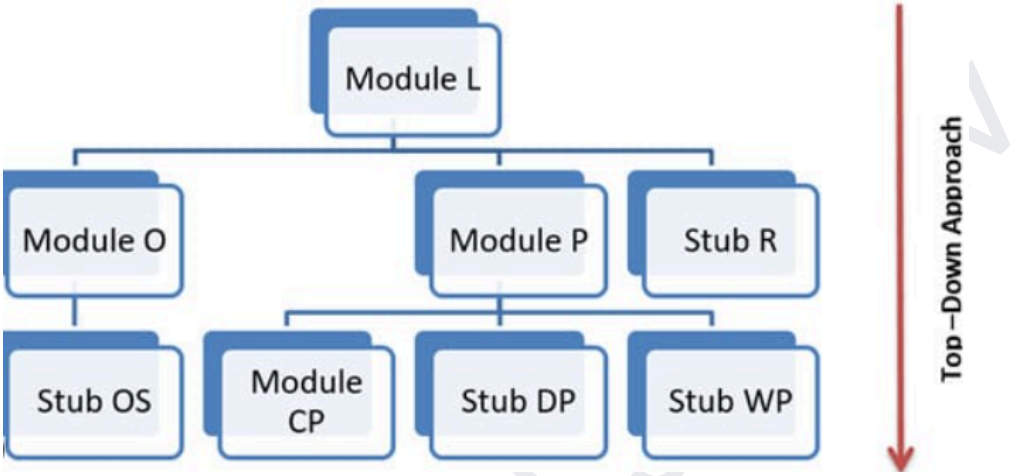
i) Top – Down ii) Bottom up iii) Sandwich

i) Top – Down integration

Testing is performed from modules present at top

to modules which are present at the bottom.

In the top down approach, stubs are used for testing.



Advantages

Identification of bug is easy

Critical modules are tested on priority basis so critical designing defects could be found and fixed early.

Disadvantages

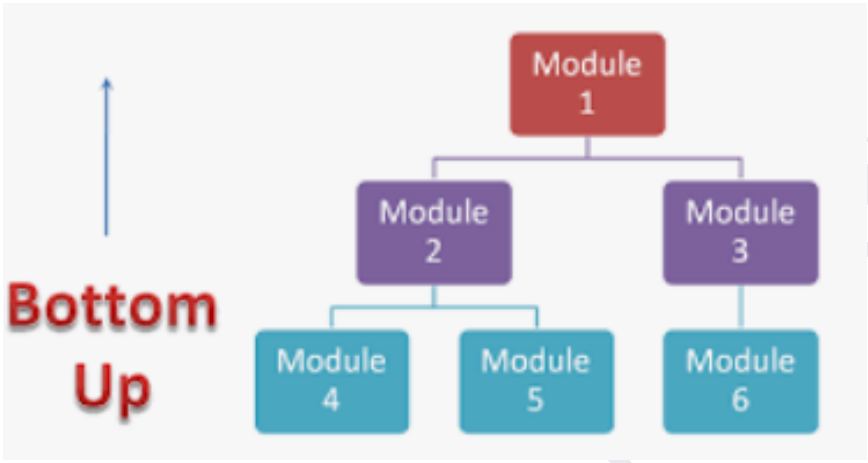
Top down testing requires many stubs because for replacing lower level modules there is a need for stubs.

Modules at lower level are tested insufficiently because of lack of time.

ii) Bottom up integration

In the bottom up approach, every module present at lower level is tested with modules present at higher levels until all modules are tested.

Drivers are used while performing bottom up incremental testing.



Advantages

Finding defect is easy

After developing one module, testing is started. No need to wait for all modules to be developed unlike the Big Bang approach.

Disadvantages

Critical modules which are present at the top level of software architecture that control the flow of software are tested last and may be defects that occur in critical modules.

iii) Sandwich integration

Consists of a combination of both top-down and bottom-up integration.

Occurs both at the highest level modules and also at the lowest level modules.

Proceeds using functional groups of modules, with each group completed before the next.

High and low-level modules are grouped based on the control and data processing they provide for a specific program feature.

Integration within the group progresses in alternating steps between the high and low level modules of the group.

When integration for a certain functional group is complete, integration and testing moves onto the next group.

Recap the advantages of both types of integration while minimizing the need for drivers and stubs.

Requires a disciplined approach so that integration doesn't tend towards the "big bang" scenario.

Smoke Testing

Smoke testing, also called build verification testing or confidence testing, is a software testing method that is used to determine if a new software build is ready for the next testing phase.

This testing method determines if the most crucial functions of a program work but does not delve into finer details.

As a preliminary check of software, smoke testing finds basic and critical issues in an application before more in-depth testing is done

The goal of smoke testing is to discover simple but severe failures using test cases that cover the most important functionalities of a software. Smoke tests are performed by QA teams using a minimal set of tests on each build that focuses on software functionality.