CHAPTER 1
INTRODICTION TO SOFTWARE TESTING

SDLC(Software Development Lifecycle):-

It is a systematic process for building software that ensures the quality and correctness of the software built. SDLC process aims to produce high-quality software that meets customers need and expectations. The system development should be complete in the pre-defined time frame and cost. SDLC consists of a detailed plan which explains how to plan, build, and maintain specific software. Every phase of the SDLC life cycle has its own process and deliverables that feed into the next phase.

SDLC phases are:-

1. **Requirement Gathering and Analysis**
2. **Design**
3. **Coding**
4. **Testing**
5. **Installation/Deployment**
6. **Maintenance**

CODING:-

Once the system design phase is over, the next phase is coding. In this phase, developers start build the entire system by writing code using the chosen programming language. In the coding phase, tasks are divided into units or modules and assigned to the various developers. It is the longest phase of the Software Development Life Cycle process. Coding, is the phase when your team works on the actual software and translates software specs and requirements into computer code. Physical software specifications are transformed into a solid working and reliable solution. Developers(coder/programmer) collect and create source code and BUILD(**.**exe file)

TESTING:-

Once the software is complete, and it is deployed in the testing environment. The testing team starts testing the functionality and Non-functionality of the entire system. This is done to verify that the entire application works according to the customer requirement.

During this phase, testing team may find some bugs/defects which they communicate to developers. The development team fixes the bug /defect and send

back to testers for a re-test. This process continues until the software is defect/bug-less, stable, and working according to the business needs of that system. Here Testers create several documents such as:-

i. Test Plan
ii. Test Scenario
iii. Test Case
iv. Defect Report
v. Test Execution Log
vi. Test Metrics
vii. Test Suites
viii. Test Procedure
ix. Test Execution Schedule
x. Test Summary Report
xi. Test Progress Report etc.....

Installation/Deployment:-

Once the s/w testing is been completed, the tested s/w is installed at the client/customer side (i.e at the client environment) and this is what happening in this phase.

Installation is done by the installation Team/Deployment team. This team is created by the project manager which consists of selected developers and testers . This team creates a documents called as User Manual/User Guide which is provided to customer for the understanding purpose of the s/w

Maintainence:-

Once the system is deployed, and customers start using the developed system, and there may occurs some issues/changes such as:-there are 2 types of changes

1) **Adaptive change:**
    a. Enhancement - Adding some new features into the existing software
2) **Corrective changes:**
    a. Bug fixing/defect fixing - bugs are reported because of some scenarios which are not tested at all
    b. Upgrade /modification- Upgrading the application to the newer versions of the Software/ modifying the existing feature in the application
    c. Removing- Deleting some existing feature in the s/w

Maintenance is done by **Maintenance team/Configuration Management team.** This team is created by the project Manager which consists of project manager himself along with BA, S/W designer, Developer, testers etc..

Different SDLC Models:-

1. Waterfall Model:-

(a) Introduction:-

- It is common and classical Life cycle model.
- The Waterfall Model was first Process Model to be introduced.
- It is also referred to as a linear-sequential life cycle model.
- It is very simple to understand and use.
- Detailed documentation in each phase before proceeding to next phase.
- Project is completely dependent on the members of team rather than client feedback.
- At the end of each phase, a review takes place to determine if the project is on the right path and whether or not to continue or discard the project.

(b) When to use the waterfall model

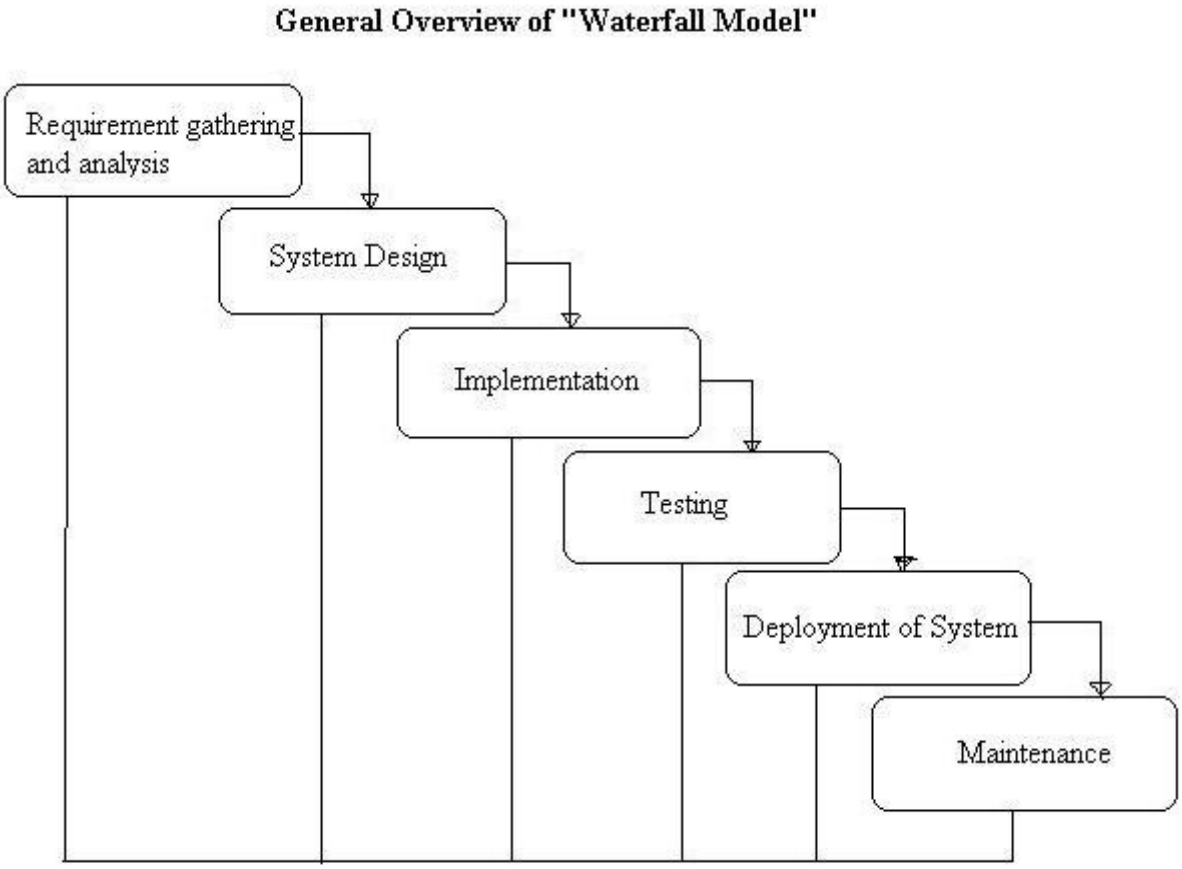
- This model is used only when the requirements are very well known, clear and fixed.
- Product definition is stable.
- Technology is understood.
- There are no ambiguous requirements.
- Ample resources with required expertise are available freely.
- The project is short.

(c) Advantages of waterfall model

- Simple and easy to understand and use.
- It is easy to manage due to the rigidity of the model – each phase has specifc deliverables and a review process.
- It works well for smaller projects where requirements are clearly defned and very well understood.

(d) Disadvantages of waterfall model

- Once an application is in the testing stage, it is very difficult to go back and change something that was not well in the previous stages.
- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing and complex and object-oriented projects.



2. Incremental Model (Multi-Waterfall Model):-

(a)Introduction:-

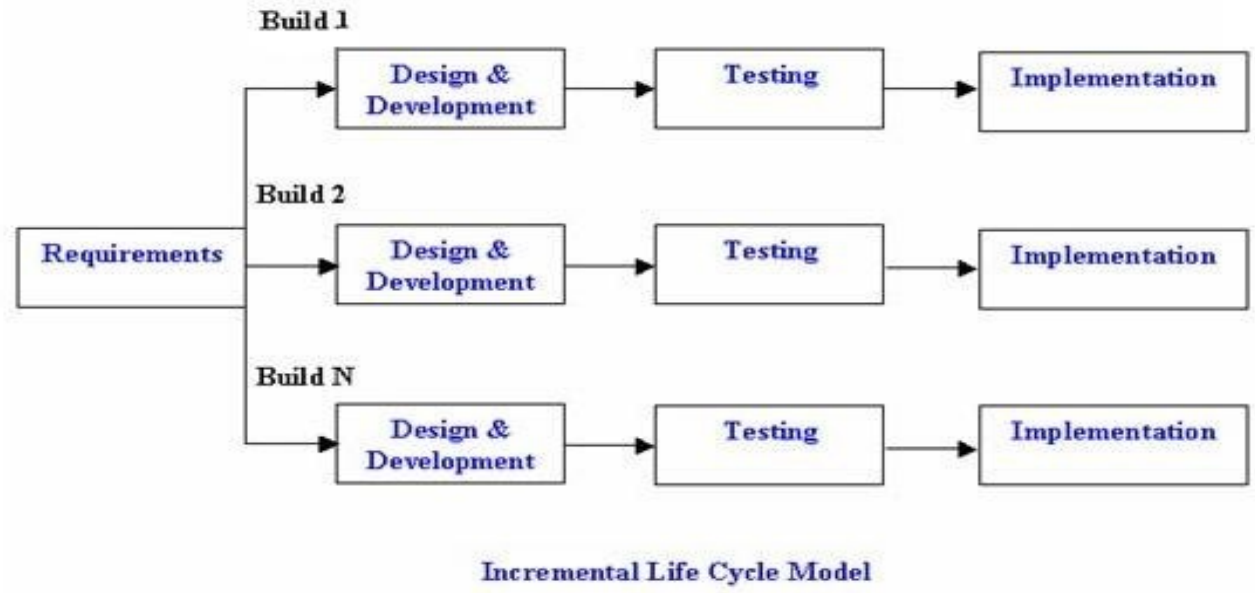
- In incremental model the whole requirement is divided into various builds.
- Multiple development cycles take place here, making the life cycle a “multi-waterfall” cycle.
- Cycles are divided up into smaller, more easily managed modules.
- In this model, each module passes through the requirements, design, implementation and testing phases.
- It doesn't at least start with the full specification of requirement.
- Instead development begin by specifying and implementing just as per high priority requirement of the s/w.
- A working version of software is produced during the first module, so you have working software early on during the software life cycle.
- Each subsequent release of the module adds function to the previous release. The process continues till the complete system is achieved.

(b) Advantages of Incremental model

- Generates working software quickly and early during the software life cycle.
- This model is more flexible – less costly to change scope and requirements.
- It is easier to test and debug during a smaller iteration.
- In this model customer can respond to each built.
- Lowers initial delivery cost.
- Easier to manage risk because risky pieces are identified and handled during it's iteration.

(c) Disadvantages of Incremental model

- Needs good planning and design.
- Needs a clear and complete defnition of the whole system before it can be broken down and built incrementally.
- Total cost is higher than waterfall.



Incremental Life Cycle Model

3. Spiral Model :-

(a) Introduction:-

- The spiral model has four phases: Planning, Risk Analysis, Engineering and Evaluation.
- A software project repeatedly passes through these phases in iterations (called Spirals in this model).
- The baseline spiral, starting in the planning phase, requirements are gathered and risk is assessed.
- Each subsequent spirals builds on the baseline spiral.

(b) **Advantages of Spiral model**

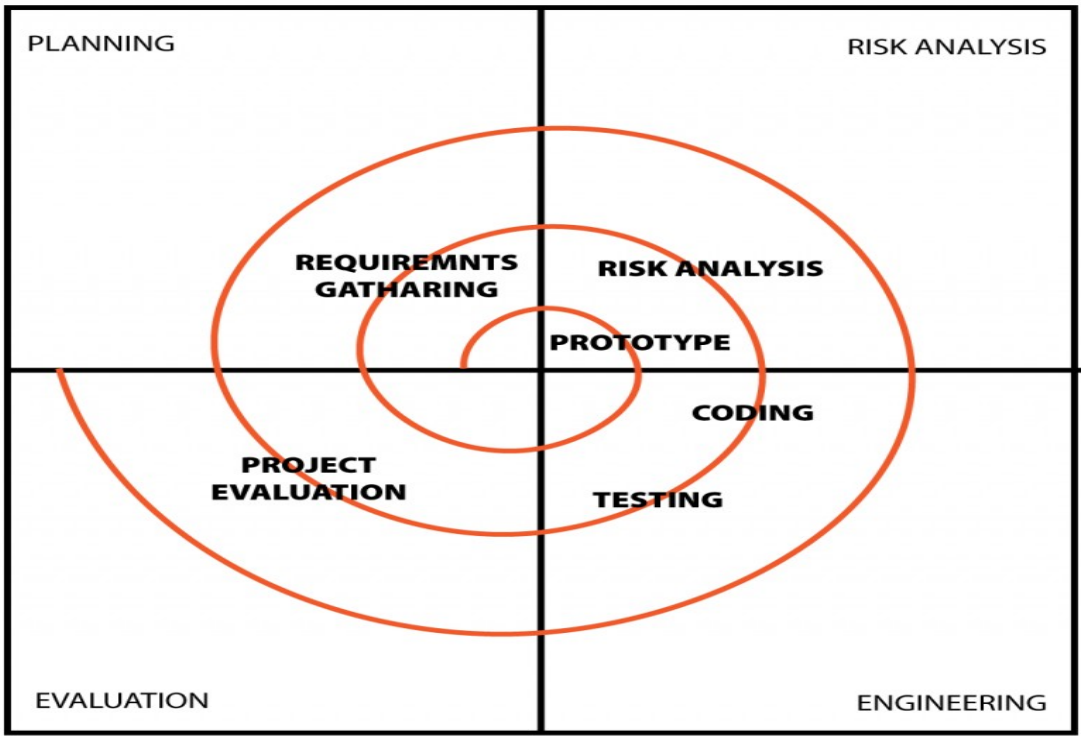
- High amount of risk analysis hence, avoidance of Risk is enhanced.
- Good for large and mission-critical projects.
- Strong approval and documentation control.
- Additional Functionality can be added at a later date.
- Software is produced early in the software life cycle.

(c) **Disadvantages of Spiral model**

- Can be a costly model to use.
- Project's success is highly dependent on the risk analysis phase.
- Doesn't work well for smaller projects.

(d) **When to use Spiral model**

- When costs and risk evaluation is important
- For medium to high-risk projects
- Users are unsure of their needs
- Requirements are complex
- Significant changes are expected.



AGILE:-

- Agile is an umbrella term for several iterative and incremental software development approaches, with each of those variations being its own Agile framework. (Framework means a set of protocol defned for doing a method and process)
- Software is developed in incremental, rapid cycles. This results in small incremental releases with each release building on previous functionality.
- Each release is thoroughly tested to ensure software quality is maintained. It is used for time critical applications.
- Extreme Programming (XP) is currently one of the most well known agile development life cycle model
- The most popular Agile frameworks include Kanban, Scrum, Extreme Programming etc.

➔ **Manifesto for Agile Software Development**

We are uncovering better ways of developing software by doing it and helping others do it.Through this work we have come to value:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Advantages of Agile model:

- Customer satisfaction by rapid, continuous delivery of useful software.
- People and interactions are emphasized rather than process and tools. Customers, developers and testers constantly interact with each other.
- Working software is delivered frequently (weeks rather than months).
- Face-to-face conversation is the best form of communication.
- Close, daily cooperation between business people and developers.
- Continuous attention to technical excellence and good design.
- Regular adaptation to changing circumstances.
- Even late changes in requirements are welcomed

Disadvantages of Agile model:

- In case of some software deliverables, especially the large ones, it is difficult to assess the effort required at the beginning of the software development life cycle.
- There is lack of emphasis on necessary designing and documentation.
- The project can easily get taken off track if the customer representative is not clear what final outcome that they want.

- It is not useful for small development projects.
- There is a lack of intensity on necessary designing and documentation.
- It requires an expert project member to take crucial decisions in the meeting.
- Cost of Agile development methodology is slightly more as compared to other development methodology.

When to use Agile model:-

- When new changes are needed to be implemented. The freedom agile gives to change is very important. New changes can be implemented at very little cost because of the frequency of new increments that are produced.
- To implement a new feature the developers need to lose only the work of a few days, or even only hours, to roll back and implement it.
- Unlike the waterfall model in agile model very limited planning is required to get started with the project. Agile assumes that the end users' needs are ever changing in a dynamic business and IT world. Changes can be discussed and features can be newly effected or removed based on feedback. This effectively gives the customer the finished system they want or need.
- Both system developers and stakeholders alike, find they also get more freedom of time and options than if the software was developed in a more rigid sequential way. Having options gives them the ability to leave important decisions until more or better data or even entire hosting programs are available; meaning the project can continue to move forward without fear of reaching a sudden standstill.

AGILE- Scrum Methodology :-

Scrum:-

What Are the Components of Agile Scrum Development?

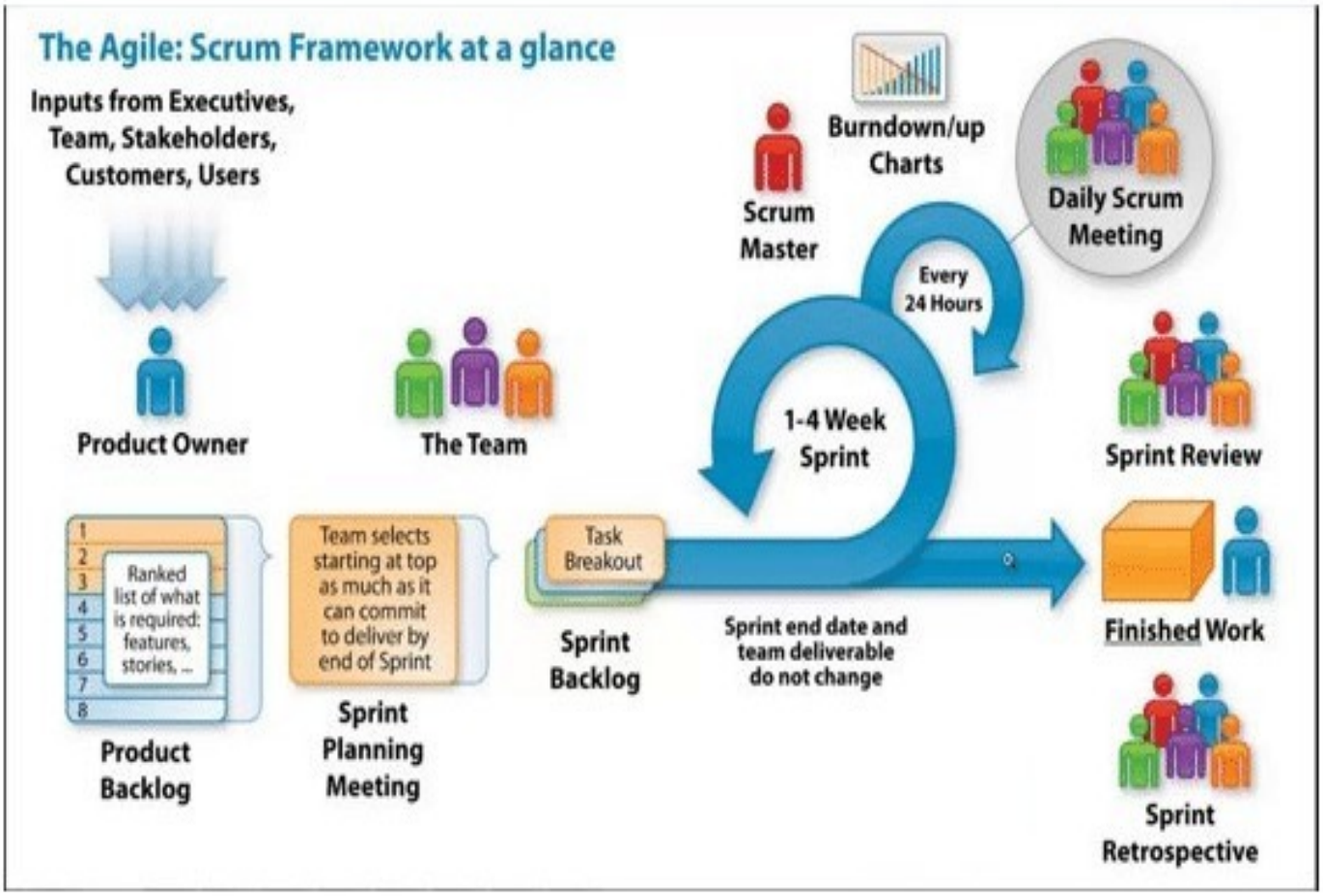
The Scrum methodology is defined by team roles, events (ceremonies), artifacts, and rules.

1. **The Scrum Team:-** Scrum teams are typically composed of 7 +/- 2 members and have no team leader to delegate tasks or decide how a problem is solved. The team as a unit decides how to address issues and solve problems. Each member of the Scrum team is an integral part of the solution and is expected to carry a product from inception to completion. There are three key roles in a **Scrum team:-**
   - The Product Owner
   - The Scrum Master
   - The Development Team

2. **Scrum Events (Ceremonies):-**
   - The Sprint
   - Sprint Planning meeting
   - The Daily Scrum Stand-up meeting
   - The Sprint Review
   - The Sprint Retrospective

3. **Scrum Artifacts :-**
   - Product Backlog
   - Sprint Backlog
   - Product Increment

4. **Scrum Rules :-**The rules of agile Scrum should be completely up to the team and governed by what works best for their processes. The best agile coaches will tell teams to start with the basic scrum events listed above and then inspect and adapt based on your team's unique needs so there is continuous improvement in the way teams work together.


The Agile: Scrum Framework at a glance
Inputs from Executives, Team, Stakeholders, Customers, Users
Burndown/up Charts
Scrum Master
Daily Scrum Meeting
Every 24 Hours
Product Owner
The Team
1-4 Week Sprint
Sprint Review
1
2
3
4
5
6
7
8
Ranked list of what is required: features, stories, ...
Team selects starting at top as much as it can commit to deliver by end of Sprint
Task Breakout
Sprint Backlog
Sprint end date and team deliverable do not change
Finished Work
Product Backlog
Sprint Planning Meeting
Sprint Retrospective

STLC:-

STLC stands for Software Testing Life Cycle. STLC is a sequence of different activities performed by the testing team to ensure the quality of the software or the product. It is the sequence of activities carried out by the testing team from the beginning of the project till the end of the project. A good software tester is expected to have good knowledge of the STLC lifecycle and its activities.

- STLC is an integral part of Software Development Life Cycle (SDLC). But, STLC deals only with the testing phases.
- STLC starts as soon as requirements are defined or SRD/SRS (Software Requirement Document / Software Requirement Specification) is shared by stakeholders.
- STLC provides a step-by-step process to ensure quality software.
- In the early stage of STLC, while the software or the product is developing, the tester can analyze and define the scope of testing, entry and exit criteria and also the Test Cases. It helps to reduce the test cycle time along with better quality.
- As soon as the development phase is over, the testers are ready with test cases and start with execution. This helps to find bugs in the initial phase.

There 5 phases is STLC and they are:-

1. **TEST PLANNING AND CONTROL**
2. **TEST ANALYSIS AND DESIGN**
3. **TEST IMPLEMENTATION AND EXECUTION**
4. **EVALUATION OF EXIT CRITERIA AND REPORTING**
5. **TEST CLOSURE ACTIVITY**

1. ## TEST PLANNING AND CONTROL:-
    a. ### Test Planning:-

    Test Planning phase starts soon after the completion of the Requirement Analysis phase. Test Planning is an activity of defining the objectives of testing and the specification of test activities in order to meet the objectives and mission. During test planning, we make sure that we understand goals of customer, stakeholders and project. We also analyse the risks which testing is intended to address. The main task of planning is to determine the test strategy or test approach. Test planning is an activity which is done by Sr. Tester/Test Manager/Test Lead/Test Coaches etc..Here the Sr. Tester is going to create a document called as “Test Plan document”. It is created with help of a standard template i.e. (IEEE829 Test plan standard template)- will described in detail in 4th chapter.

Test planning major tasks are:

- Determine the scope and risks and identify the objective of testing.
- Determine the test approach.
- Implement the test policy and/or test strategy.
- Determine the required test resources
- Schedule test analysis and design tasks, test implementation, execution and evaluation.
- Determine the exit criteria.

b. **Test Control:-**

Test Control is the ongoing activity of comparing actual progress against the plan. We need to control and measure progress against the plan. We need to control and measure progress against the plan to compare actual progress against the planned progress, and we should report to the project manager and customer on the current status of testing, including any changes or deviations from the plan. We'll need to take actions where necessary to meet the objectives of the project. Such actions may entail changing our original plan, which often happens.

Test control has following major tasks:

- Measure and analyse the results of reviews and testing
- Monitor and document progress, test coverage and exit criteria
- Provide Information on Testing
- Initiate corrective actions
- Make decisions

Deliverables (Outcome) of Test Planning and control phase are:-

- Test Plan document created by Sr.Tester/TL/TM
- Test Strategy document
- Best suited Testing Approach

2. **TEST ANALYSIS AND DESIGN:-**
    a. **Test Analysis:-**
    Test Analysis is process of analyzing all the documents from which the requirements of a component or system can be inferred and defining test objectives. It covers "**WHAT**" is to be tested in the form of test conditions and can start as soon as the basis for testing is established for each test level.
    b. **Test Design:-**
    Test design is a process of defining the test conditions and creating the Test Scenario document. Test analysis and design is done by Sr. Tester/TM/TL

Deliverables (Outcome) of Test analysis and design phase are:-

- Test Scenario document created by Sr. Tester/TM/TL

-

3. **TEST IMPLEMENTATION AND EXECUTION:-**
    a. **Test Implementation:-**
    Test Implementation is an item or event of a component or system that could be verified by one or more test cases (ex: function, transaction, feature, etc.).It Covers "**HOW"** something is to be tested by identifying test cases with step wise elaboration for the test conditions (from Test Analysis and design phase).This phase can start for a given Test Level once Test Conditions are identified and enough information is available to enable the production of Test Cases. In other words, a test case is a set of input values, execution preconditions, expected results and execution post-conditions, developed for a particular objective or test condition, such as to exercise a particular program path or to verify compliance with a specific requirement.

    This Phase is done by Sr. Tester/TL/TM (i.e. test case is developed by Sr.Tester)

    b. **Test Execution:-**
    Test execution is the process of executing the software and comparing the expected and actual results. This Phase is done Jr. Tester. Here the Jr. Tester executes the test case document and finds the defect from the Software(build) and they creates documents such as Defect Report and Execution Log.
    The following points need to be considered for Test Execution.

- In this phase, the testing team performs actual validation of Software based on prepared test cases and compares the stepwise result with the expected result.
- The entry criteria of this phase is completion of the Test Plan and the Test Cases Development phase, the test data should also be ready.
- The validation of Test Environment setup is always recommended through smoke testing before officially entering the test execution.
- The exit criteria requires the successful validation of all Test Cases; Defects should be closed or deferred; test case execution and defect summary report should be ready.

Activities for Test Execution:-

The objective of this phase is real time validation of s/w before moving on to production/release. To sign off from this phase, the QA team performs different types of testing to ensure the quality of product. Along with this defect reporting and retesting is also crucial activity in this phase.

Deliverables (Outcome) of Test Implementation and Execution phase are:-

- Test Case document created by Sr. Tester.
- Defect Report created by Jr. Tester
- Execution Log created by Jr. Tester

4. **EVALUATION OF EXIT CRITERIA AND REPORTING:-**
   a. **Evaluation of exit criteria:-**

      Exit criteria is set of agreed conditions with stakeholders based on which you can officially mark the testing process to be completed for a particular test level. Exit criteria should be set for each test level. In exit criteria evaluation we assess the test execution against the defined and agreed exit criteria for a particular test level. Based on this evaluation we can decide if we have actually done enough testing for that level to mark it officially complete.

Some common points that are mostly present in exit criteria are as follows:

- All critical, high and medium priority test cases are executed.
- No blocker, critical or high priority defects are outstanding.
- All medium and low priority defects are triaged and agreed to be differed or fixed.
- Agreed fixes for medium and low priority defects completed and retested.
- All the test documentation like test plan, test cases are documented and up to date in the test management tool.

The main tasks for evaluating exit criteria are as follows:

- Checking the test logs against the defined exit criteria. For example, checking the test execution progress for all the critical, major and medium test cases. Checking the status of all outstanding defects and how many needs to be fixed and retested to fulfill the exit criteria.
- Secondly we need to assess if there are more tests that need to be executed because of the product quality concerns (i.e. you found more than expected integration issues). You might want to add more tests for execution to lower the quality risks specified in the the exit criteria. Sometimes you might also need to modify the exit criteria with agreement from stakeholders if it had very strict criteria set initially but at current stage of project those criteria pose minimum risk so that exit criteria is met.

b. **Reporting:-**

Here we write the test summary report .Test summary report contains the summarized description of all the activities of testing process and it mention details about the defect and testing status of the process.
When you are evaluating the exit criteria it's not only Development team and Test team need to know the outcome but also the broader set of stakeholders need to know what happened in testing levels so that they can make decisions about the software. For example, is it good to start the next testing level or if all the test levels have completed can the software go live in production.

Deliverables (Outcome) of Evaluation of Exit Criteria and Reporting phase are:-

- Test Summary report

5. **TEST CLOSURE ACTIVITY:-**

   Test closure activities are those activities which are performed at the end of the testing process. These are usually performed after the product is delivered. After determining that test execution has been complete the data from completed test activities need to be collected and consolidated. You need to analyze the data to find out facts and numbers about the testing activities during project cycle. Test closure activities are done mostly after the product is delivered but there are other instances as well where test closure is done like, if project got cancelled or after maintenance release is done. Here in this phase, all the documents which were created throughout the project cycle is collected and consolidated together and it is stored in the database. So that if in future if there is any similar projects comes up then we can reuse the test documents for that similar project. This will make sure that mistakes should not be repeated which happened before . This phase is also done for knowledge sharing and the learning purpose

-

Methods of Testing:-

1. **Black Box Testing:-**The technique of testing without having any knowledge of the interior workings of the application is called black-box testing. The tester is oblivious to the system architecture and does not have access to the source code. Typically, while performing a black-box test, a tester will interact with the system's user interface by providing inputs and examining outputs without knowing how and where the inputs are worked upon.

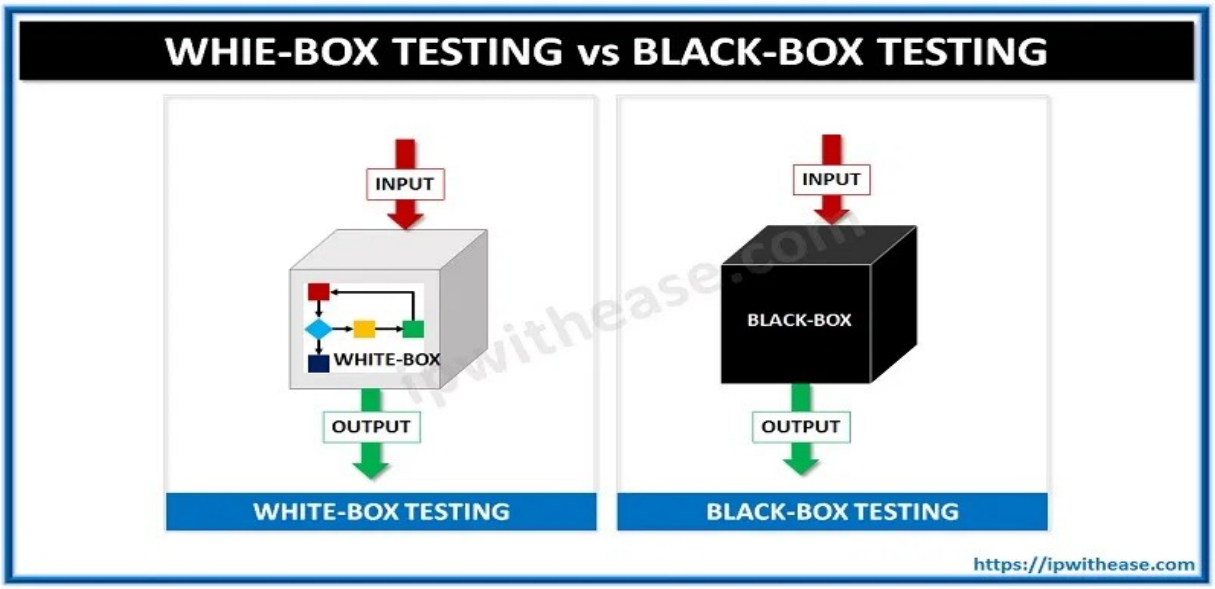
Black-Box Testing Techniques:-

- Equivalence Class partitioning.
- Boundary value analysis.
- Error Guessing.

2. **White Box Testing:-** White-box testing is the detailed investigation of internal logic and structure of the code. White-box testing is also called **glass testing** or **open-box testing**. In order to perform **white-box** testing on an application, a tester needs to know the internal workings of the code. The tester needs to have a look inside the source code and find out which unit/chunk of the code is behaving inappropriately.

White-Box Testing Techniques:-

- Statement coverage.
- Decision coverage.
- Condition Coverage
- Path Coverage
- Control testing
- Data Flow testing etc..



Differences between Black Box Testing vs White Box Testing:

Black Box Testing	White Box Testing
It is a way of software testing in which the internal structure or the program or the code is hidden and nothing is known about it.	It is a way of testing the software in which the tester has knowledge about the internal structure r the code or the program of the software.
It is mostly done by software testers.	It is mostly done by software developers.
It is functional test of the software.	It is structural test of the software.
This testing can be initiated on the basis of requirement specifications document.	This type of testing of software is started after detail design document.
No knowledge of programming is required.	It is mandatory to have knowledge of programming.
It is the behavior testing of the software.	It is the logic testing of the software.
It is applicable to the higher levels of testing of software.	It is generally applicable to the lower levels of software testing.
It is also called closed or Functional testing.	It is also called as Clear box , Glass box , Transparent testing.
It is least time consuming.	It is most time consuming.
Example: search something on google by using keywords	**Example:** by input to check and verify loops

Black-Box Testing Techniques:-

- Equivalence Class partitioning:- Group the condition into Equal Class Partition and select any random value from each partition. (Note:- there are 2 classes, they are 1.Valid Class 2.Invalid Class)
- Boundary value analysis:- Group the condition into Equal Class Partition and select Maximum and Minimum value from from the edges of valid class partition or +1 and -1 to the edges of valid class partition.
- Error Guessing:- It is an Experienced based techniques where testing of software is done based on the testers skills and knowledge and experience

Defect:- the difference or variance between the excepted result and the actual result is called as the defect. There are 2 types of defect

- **Functional defect**
- **Non-Functional defect**