SOFTWARE TESTING

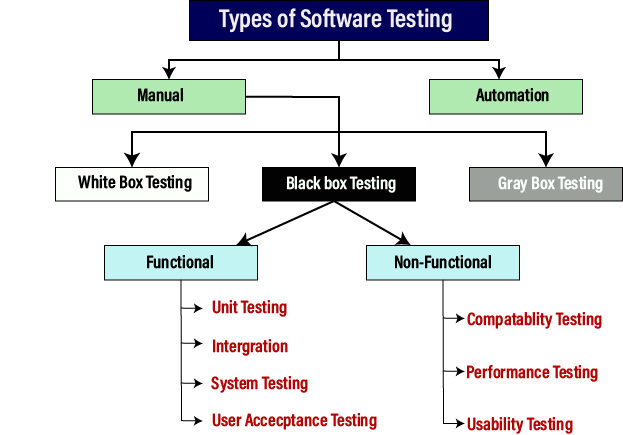
Software testing is a process of identifying the correctness of software by considering its all attributes (Reliability, Scalability, Portability, Re-usability, Usability) and evaluating the execution of software components to find the software bugs or errors or defects.

TYPES OF SOFTWARE TESTING:

1. MANUAL TESTING:

* Black box testing
* White box testing
* Grey box testing

1. AUTOMATION TESTING



Manual testing

The process of checking the functionality of an application as per the customer needs without taking any help of automation tools is known as manual testing. While performing the manual testing on any application, we do not need any specific knowledge of any testing tool, rather than have a proper understanding of the product so we can easily prepare the test document.

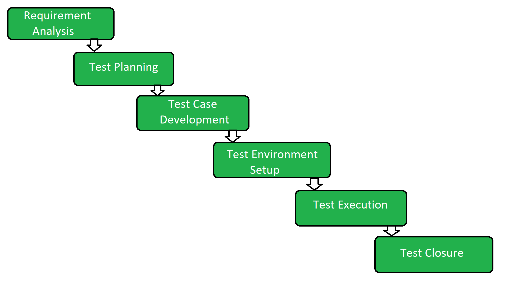
Manual testing can be further divided into three types of testing, which are as follows:

* **White box testing**
* **Black box testing**
* **Grey box testing**

### Automation testing

Automation testing is a process of converting any manual test cases into the test scripts with the help of automation tools, or any programming language is known as automation testing. With the help of automation testing, we can enhance the speed of our test execution because here, we do not require any human efforts. We need to write a test script and execute those scripts.

STLC DIAGRAM :



## **Phases of STLC**

**1. Requirement Analysis:**Requirement Analysis is the first step of the Software Testing Life Cycle (STLC). In this phase quality assurance team understands the requirements like what is to be tested. If anything is missing or not understandable then the quality assurance team meets with the stakeholders to better understand the detailed knowledge of requirements.

**The activities that take place during the Requirement Analysis stage include:**

* Reviewing the software requirements document (SRD) and other related documents
* Interviewing stakeholders to gather additional information
* Identifying any ambiguities or inconsistencies in the requirements
* Identifying any missing or incomplete requirements
* Identifying any potential risks or issues that may impact the testing process

**2. Test Planning:**Test Planning is the most efficient phase of the software testing life cycle where all testing plans are defined. In this phase manager of the testing, team calculates the estimated effort and cost for the testing work. This phase gets started once the requirement-gathering phase is completed.

**The activities that take place during the Test Planning stage include:**

* Identifying the testing objectives and scope
* Developing a test strategy: selecting the testing methods and techniques that will be used
* Identifying the testing environment and resources needed
* Identifying the test cases that will be executed and the test data that will be used
* Estimating the time and cost required for testing
* Identifying the test deliverables and milestones
* Assigning roles and responsibilities to the testing team
* Reviewing and approving the test plan

**3. Test Case Development:**The test case development phase gets started once the test planning phase is completed. In this phase testing team notes down the detailed test cases. The testing team also prepares the required test data for the testing. When the test cases are prepared then they are reviewed by the quality assurance team.

**The activities that take place during the Test Case Development stage include:**

* Identifying the test cases that will be developed
* Writing test cases that are clear, concise, and easy to understand
* Creating test data and test scenarios that will be used in the test cases
* Identifying the expected results for each test case
* Reviewing and validating the test cases
* Updating the requirement traceability matrix (RTM) to map requirements to test cases

## 4.Test Environment Setup:

For the test environment, a key area to set up includes

* System and applications
* Test data
* Database server
* Front-end running environment
* Client operating system
* Browser
* Hardware includes Server Operating system
* Network
* Documentation required like reference documents/configuration guides/installation guides/ user manuals

**5. Test Execution:**After the test case development and test environment setup test execution phase gets started. In this phase testing team starts executing test cases based on prepared test cases in the earlier step.

**The activities that take place during the test execution stage of the Software Testing Life Cycle (STLC) include:**

* **Test execution:** The test cases and scripts created in the test design stage are run against the software application to identify any defects or issues.
* **Defect logging:** Any defects or issues that are found during test execution are logged in a defect tracking system, along with details such as the severity, priority, and description of the issue.
* **Test data preparation:** Test data is prepared and loaded into the system for test execution
* **Test environment setup:** The necessary hardware, software, and network configurations are set up for test execution
* **Test execution:**The test cases and scripts are run, and the results are collected and analyzed.
* **Test result analysis:** The results of the test execution are analyzed to determine the software’s performance and identify any defects or issues.
* **Defect retesting:** Any defects that are identified during test execution are retested to ensure that they have been fixed correctly.
* **Test Reporting:** Test results are documented and reported to the relevant stakeholders.

It is important to note that test execution is an iterative process and may need to be repeated multiple times until all identified defects are fixed and the software is deemed fit for release.

**6. Test Closure:**Test closure is the final stage of the Software Testing Life Cycle (STLC) where all testing-related activities are completed and documented. The main objective of the test closure stage is to ensure that all testing-related activities have been completed and that the software is ready for release.

* **Test summary report:** A report is created that summarizes the overall testing process, including the number of test cases executed, the number of defects found, and the overall pass/fail rate.
* **Defect tracking:** All defects that were identified during testing are tracked and managed until they are resolved.
* **Test environment clean-up:**The test environment is cleaned up, and all test data and test artefact are archived.
* **Test closure report:** A report is created that documents all the testing-related activities that took place, including the testing objectives, scope, schedule, and resources used.
* **Knowledge transfer:** Knowledge about the software and testing process is shared with the rest of the team and any stakeholders who may need to maintain or support the software in the future.
* **Feedback and improvements:** Feedback from the testing process is collected and used to improve future testing processes

# Bug Life Cycle in Software Testing:

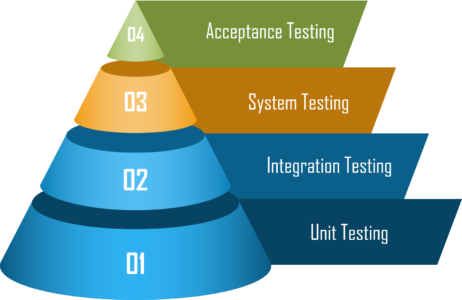
**Defect Life Cycle** or Bug Life Cycle in software testing is the specific set of states that defect or bug goes through in its entire life. The purpose of Defect life cycle is to easily coordinate and communicate current status of defect which changes to various assignees and make the defect fixing process systematic and efficient.

PHASES OF BUG LIFE CYCLE:



* **New:** When a new defect is logged and posted for the first time. It is assigned a status as NEW.
* **Assigned:** Once the bug is posted by the tester, the lead of the tester approves the bug and assigns the bug to the developer team
* **Open**: The developer starts analyzing and works on the defect fix
* **Fixed**: When a developer makes a necessary code change and verifies the change, he or she can make bug status as “Fixed.”
* **Pending retest**: Once the defect is fixed the developer gives a particular code for retesting the code to the tester. Since the software testing remains pending from the testers end, the status assigned is “pending retest.”
* **Retest**: Tester does the retesting of the code at this stage to check whether the defect is fixed by the developer or not and changes the status to “Re-test.”
* **Verified**: The tester re-tests the bug after it got fixed by the developer. If there is no bug detected in the software, then the bug is fixed and the status assigned is “verified.”
* **Reopen**: If the bug persists even after the developer has fixed the bug, the tester changes the status to “reopened”. Once again the bug goes through the life cycle.
* **Closed**: If the bug is no longer exists then tester assigns the status “Closed.”
* **Duplicate**: If the defect is repeated twice or the defect corresponds to the same concept of the bug, the status is changed to “duplicate.”
* **Rejected**: If the developer feels the defect is not a genuine defect then it changes the defect to “rejected.”
* **Deferred**: If the present bug is not of a prime priority and if it is expected to get fixed in the next release, then status “Deferred” is assigned to such bugs
* **Not a bug**: If it does not affect the functionality of the application then the status assigned to a bug is “Not a bug”.

LEVELS OF TESTING

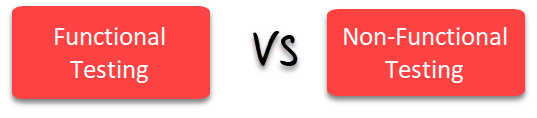


1. [Unit Testing](https://www.geeksforgeeks.org/unit-testing-software-testing/)**:**  
   In this type of testing, errors are detected individually from every component or unit by individually testing the components or units of software to ensure that if they are fit for use by the developers. It is the smallest testable part of the software.
2. [Integration Testing](https://www.geeksforgeeks.org/software-engineering-integration-testing/)**:**  
   In this testing, two or more modules which are unit tested are integrated to test i.e. technique interacting components and are then verified if these integrated modules work as per the expectation or not and interface errors are also detected.
3. [System Testing](https://www.geeksforgeeks.org/system-testing/)**:**  
   In system testing, complete and integrated Software’s are tested i.e. all the system elements forming the system is tested as a whole to meet the requirements of the system.
4. [Acceptance Testing](https://www.geeksforgeeks.org/acceptance-testing-software-testing/)**:**  
   It is a kind of testing conducted to ensure whether the requirement of the users are fulfilled prior to its delivery and the software works correctly in the user’s working environment.

## Conclusion:

* A level of software testing is a process where every unit or component of a software/system is tested.
* The primary goal of system testing is to evaluate the system’s compliance with the specified needs.
* In Software Engineering, four main levels of testing are Unit Testing, Integration Testing, System Testing and Acceptance Testing.

## Difference between Functional Testing and Non Functional Testing



| Parameters | Functional | Non-functional testing |
| --- | --- | --- |
| **Execution** | It is performed before non-functional testing. | It is performed after the functional testing. |
| **Focus area** | It is based on customer’s requirements. | It focusses on customer’s expectation. |
| **Requirement** | It is easy to define functional requirements. | It is difficult to define the requirements for  non-functional testing. |
| **Usage** | Helps to validate the behaviour of the application. | Helps to validate the performance of the application. |
| **Objective** | Carried out to validate software actions. | It is done to validate the performance of the software. |
| **Requirements** | Functional testing is carried out using the functional specification. | This kind of testing is carried out by performance  Specifications. |
| **Manual testing** | Functional testing is easy to execute by manual testing. | It’s very hard to perform non-functional testing  manually. |
| **Functionality** | It describes what the product does. | It describes how the product works. |
| **Example Test Case** | Check login functionality. | The dashboard should load in 2 seconds. |
| **Testing Types** | Examples of Functional Testing Types   * Unit testing * Smoke testing * User Acceptance * Integration Testing * Regression testing * Localization * Globalization * Interoperability | Examples of Non-functional Testing Types   * Performance Testing * Volume Testing * Scalability * Usability Testing * Load Testing * Stress Testing * Compliance Testing * Portability Testing * Disaster Recover Testing |

### Regression testing

* Any new change or feature added to the software can wreck its existing functionalities. Regression testing is performed every time alterations are made to check for the software’s stability and functionalities. Due to its work-intensive nature, regression testing is often automated.

Example:  A food delivery app added a function to help users add multiple promotions on top of each other. A regression test needs to be done to make sure the checkout and payment process is not affected.

### Sanity testing

* Similar to regression testing, sanity testing is conducted for a new build with minor bug fixes, or new code added. If rejected in the sanity testing phase, the build will not proceed to further testing. While regression testing checks the entire system after alterations, sanity testing targets specific areas that are affected by the new code or bug fixes only.

Example: On an e-commerce webpage, users cannot add a particular product to their cart even when the stock is available. After the issue was fixed, sanity testing is performed to ensure that the “add to cart” function is indeed working.

### Smoke testing

* When a new build is completed, it is handed to the QAs for smoke testing. In this phase, only the most critical and core functionalities are tested to ensure that they yield the intended results. As an early-stage acceptance test, smoke testing adds a verification layer to determine whether or not the new build can proceed to the next stage or needs re-work.

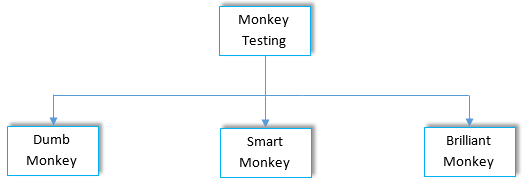
Example:  A utility company built an app with the function to report outages in customers’ homes. This function reports the address and other relevant information as well as notifies the homeowner when a dispatcher is on the way to help. Smoke testing will validate this feature on a fundamental level to assure that when an outage is reported, the correct information is sent so a dispatcher can be there on time.

## Monkey Testing

* **Monkey Testing** is a software testing technique in which the tester enters any random inputs into the software application without predefined test cases and checks the behaviour of the software application, whether it crashes or not. The purpose of Monkey testing is to find the bugs and errors in the software application using experimental techniques.

STEPS OF MONKEY TESTING

1. In Monkey Testing the tester (sometimes developer too) is considered as the ‘Monkey’
2. If a monkey uses a computer, he will randomly perform any task on the system out of his understanding
3. Just like the tester will apply random test cases on the system under test to find bugs/errors without predefining any test case
4. In some cases, Monkey Testing is dedicated to [Unit Testing](https://www.guru99.com/unit-testing-guide.html) or [GUI Testing](https://www.guru99.com/gui-testing.html) too



1. Dumb Monkey:Testers have no idea about the system and its functionality, also no assurance about the validity of test case.
2. Smart Monkey:Tester has a precise idea about system its purpose and functionality. Tester navigates through the system and gives valid inputs to perform testing.
3. Brilliant Monkey: Testers perform testing as per user’s behaviour and can specify some probabilities of bugs to have occurred.

TESTING TERMS

**QUALITY ASSURANCE:**Part of quality management focused on providing confidence that quality requirements will be fulfilled.

**QUALITY CONTROL:** It is a systematic set of processes used to ensure the quality of software products or services. The main purpose of the quality control process is ensuring that the software product meets the actual requirements by testing and reviewing its functional and non-functional requirements.

**VERIFICATION: It** is a process of checking documents, design, code, and program in order to check if the software has been built according to the requirements or not. The main goal of verification process is to ensure quality of software application, design, architecture etc.

**VALIDATION: It** is a dynamic mechanism of testing and validating if the software product actually meets the exact needs of the customer or not. The process helps to ensure that the software fulfils the desired use in an appropriate environment.

## Black Box Testing

**Black Box Testing** is a software testing method in which the functionalities of software applications are tested without having knowledge of internal code structure, implementation details and internal paths. Black Box Testing mainly focuses on input and output of software applications and it is entirely based on software requirements and specifications. It is also known as Behavioral Testing.



The above Black-Box can be any software system you want to test. For Example, an operating system like Windows, a website like Google, a database like Oracle or even your own custom application. Under Black Box Testing, you can test these applications by just focusing on the inputs and outputs without knowing their internal code implementation. Consider the following video tutorial-

**Table of Content:**

Click [here](https://www.guru99.com/faq#faq1) if the video is not accessible

## Black Box Testing Techniques

Following are the prominent[Test Strategy](https://www.guru99.com/how-to-create-test-strategy-document.html)amongst the many used in Black box Testing

* **Equivalence Class Testing:** It is used to minimize the number of possible test cases to an optimum level while maintains reasonable test coverage.
* **Boundary Value Testing:** Boundary value testing is focused on the values at boundaries. This technique determines whether a certain range of values are acceptable by the system or not. It is very useful in reducing the number of test cases. It is most suitable for the systems where an input is within certain ranges.
* **Decision Table Testing**: A decision table puts causes and their effects in a matrix. There is a unique combination in each column.

## Types of Black Box Testing

There are many types of Black Box Testing but the following are the prominent ones –

* **Functional testing** – This black box testing type is related to the functional requirements of a system; it is done by software testers.
* **Non-functional testing**– This type of black box testing is not related to testing of specific functionality, but non-functional requirements such as performance, scalability, usability.
* **Regression testing**– [Regression Testing](https://www.guru99.com/regression-testing.html) is done after code fixes, upgrades or any other system maintenance to check the new code has not affected the existing code.

## How to do BlackBox Testing in Software Engineering

Here are the generic steps followed to carry out any type of Black Box Testing.

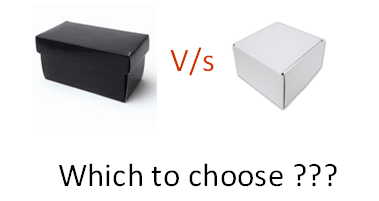
* Initially, the requirements and specifications of the system are examined.
* Tester chooses valid inputs (positive test scenario) to check whether SUT processes them correctly. Also, some invalid inputs (negative test scenario) are chosen to verify that the SUT is able to detect them.
* Tester determines expected outputs for all those inputs.
* Software tester constructs test cases with the selected inputs.
* The test cases are executed.
* Software tester compares the actual outputs with the expected outputs.
* Defects if any are fixed and re-tested.

## Tools used for Black Box Testing:

Tools used for Black box testing largely depends on the type of black box testing you are doing.

* For Functional/ Regression Tests you can use – [QTP](https://www.guru99.com/quick-test-professional-qtp-tutorial.html), [Selenium](https://www.guru99.com/selenium-tutorial.html)
* For Non-Functional Tests, you can use – [LoadRunner](https://www.guru99.com/loadrunner-v12-tutorials.html), [Jmeter](https://www.guru99.com/jmeter-tutorials.html)

## Comparison of Black Box and White Box Testing:



|  |  |
| --- | --- |
| **Black Box Testing** | **White Box Testing** |
| the main focus of black box testing is on the validation of your functional requirements. | [White Box Testing](https://www.guru99.com/white-box-testing.html) (Unit Testing) validates internal structure and working of your software code |
| Black box testing gives abstraction from code and focuses on testing effort on the software system behavior. | To conduct White Box Testing, knowledge of underlying programming language is essential. Current day software systems use a variety of programming languages and technologies and its not possible to know all of them. |
| Black box testing facilitates testing communication amongst modules | White box testing does not facilitate testing communication amongst modules |

## Black Box Testing and Software Development Life Cycle (SDLC)

Black box testing has its own life cycle called Software Testing Life Cycle ([STLC](https://www.guru99.com/software-testing-life-cycle.html)) and it is relative to every stage of Software Development Life Cycle of Software Engineering.

* **Requirement** – This is the initial stage of SDLC and in this stage, a requirement is gathered. Software testers also take part in this stage.
* **Test Planning & Analysis** – [Testing Types](https://www.guru99.com/types-of-software-testing.html) applicable to the project are determined. A[Test Plan](https://www.guru99.com/what-everybody-ought-to-know-about-test-planing.html)is created which determines possible project risks and their mitigation.
* **Design** – In this stage Test cases/scripts are created on the basis of software requirement documents
* **Test Execution**– In this stage Test Cases prepared are executed. Bugs if any are fixed and re-tested.

## What is Performance Testing?

**Performance Testing** is a software testing process used for testing the speed, response time, stability, reliability, scalability, and resource usage of a software application under a particular workload. The main purpose of performance testing is to identify and eliminate the performance bottlenecks in the software application. It is a subset of performance engineering and is also known as ***“Perf Testing”***.

The focus of Performance Testing is checking a software program’s

* **Speed** – Determines whether the application responds quickly
* **Scalability** – Determines the maximum user load the software application can handle.
* **Stability** – Determines if the application is stable under varying loads

## Types of Performance Testing

* **Load testing –** checks the application’s ability to perform under anticipated user loads. The objective is to identify performance bottlenecks before the software application goes live.

Tools used for testing

* + - Web load
    - Load ninja
    - Load runner
    - ApacheJmeter

**Stress testing –** involves testing an application under extreme workloads to see how it handles high traffic or data processing. The objective is to identify the breaking point of an application.

Tools used for testing

* + - Web load
    - Load ninja
    - Load runner
    - ApacheJmeter

**Endurance testing –** is done to make sure the software can handle the expected load over a long period of time.

Tools used for testing

* + - Web load
    - Apache J meter

**Spike testing –** tests the software’s reaction to sudden large spikes in the load generated by users.

Tools used for testing

* + - Soap UI
    - Cloud test
    - Apache J meter

**Volume testing** – Under Volume Testing large no. of. Data is populated in a database, and the overall software system’s behaviour is monitored. The objective is to check software application’s performance under varying database volumes.

Tools used for testing

* DB fit
* Hammer DB

**Scalability testing**– The objective of scalability testing is to determine the software application’s effectiveness in “scaling up” to support an increase in user load. It helps plan capacity addition to your software system.

Tools used for testing

* + - Web load
    - Appvance
    - Apache J meter

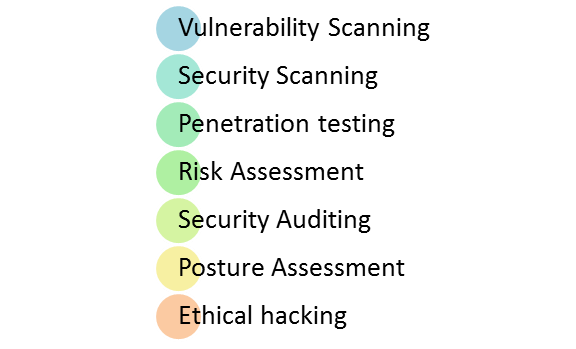
# Security Testing

* **Security Testing** is a type of [Software Testing](https://www.geeksforgeeks.org/software-testing-basics/) that uncovers vulnerabilities of the system and determines that the data and resources of the system are protected from possible intruders.
* It ensures that the software system and application are free from any threats or risks that can cause a loss.
* Security testing of any system is focused on finding all possible loopholes and weaknesses of the system which might result in the loss of information or repute of the organization.

Principle of Security Testing:

* Confidentiality
* Integrity
* Authentication
* Authorization
* Availability
* Non-repudiation

## Types of Security Testing in Software Testing



* Vulnerability Scanning: This is done through automated software to scan a system against known vulnerability signatures.
* Security Scanning: It involves identifying network and system weaknesses, and later provides solutions for reducing these risks. This scanning can be performed for both Manual and Automated scanning.
* Penetration testing: This kind of testing simulates an attack from a malicious hacker. This testing involves analysis of a particular system to check for potential vulnerabilities to an external hacking attempt.
* Risk Assessment: This testing involves analysis of security risks observed in the organization. Risks are classified as  Low, Medium and High. This testing recommends controls and measures to reduce the risk.
* Security Auditing: This is an internal inspection of Applications and Operating systems for security flaws. An audit can also be done via line by line inspection of code
* Ethical hacking: It’s hacking an Organization Software systems. Unlike malicious hackers, who steal for their own gains, the intent is to expose security flaws in the system.
* Posture Assessment: This combines Security scanning,[Ethical Hacking](https://www.guru99.com/ethical-hacking-tutorials.html)and Risk Assessments to show an overall security posture of an organization.

EXAMPLES:

* A password should be in encrypted format
* Application or System should not allow invalid users
* Check cookies and session time for application
* For financial sites, the Browser back button should not work.

## Security Testing Roles

* Hackers – Access computer system or network without authorization
* Crackers – Break into the systems to steal or destroy data
* Ethical Hacker – Performs most of the breaking activities but with permission from the owner
* Script Kiddies or packet monkeys – Inexperienced Hackers with programming language skill

TOOLS USED :

* ACUNETIX
* INTRUDER
* OWASP
* WIRESHARK
* W3af

Advantages and disadvantages of security testing

|  |  |
| --- | --- |
| **Advantages** | **Disadvantages** |
| * The best way to discover security vulnerabilities | * Requires a wide breadth of testing |
| * Many tools available | * Requires dedicated knowledge and skills |
| * Easy to automate | * Constantly evolving security vulnerabilities and threats |



# Test data Management (TDM)

* Test data management is a type of operation that ensures accurate data is ready for the testers to help them in their swanky work process.
* Test data management (TDM) accomplishes compatible test data properly and flawlessly.
* TDM provides and creates a brief automated test that can benefit users without any difficulty in the middle of their work.

Features of Test Data Management:

1. Provides required test data
2. Provides inbuilt data library
3. Allows sensitive data masking and encryption
4. Identifies data sources
5. Prepares data generation rules
6. Protection of sensitive data and many more.

Importance of Test Data Management:

1. TDM Provides high-quality software that will work effectively.
2. It will obstruct unusual bug fixes.
3. Stored the primitive data successfully
4. Decreases the risks of misplacement of the information.
5. Provides test data to application members in an accurate time.
6. Avoids higher costs.
7. Maintains security for sensitive information.
8. Keeps data always in the right position.
9. Provides easy access to the testers.
10. No extra steps for the test data by a lot of team members.

Best tools for the Test Data Management:

1. Informatica
2. LA Test Data Manager
3. LISA Solutions
4. Compuware
5. Delphix
6. Microfocus Data Express
7. IBM InfoSphere Optim

## Test Environment Management

Test environment management (TEM) is a function in a software delivery process which aids the software testing cycle by providing a validated, stable and usable test environment to execute the test scenarios or replicate bugs.

## Benefits of Test Environment Management

* Lower test environment set-up and support costs.
* Flexible and faster test environment provisioning and support services delivery.
* End-to-end environment management.
* Greater co-ordination and control of change.
* Defined and measurable outcomes.

PHASES OF SDLC

**1. Requirements gathering and analysis:** This phase involves gathering information about the software requirements from stakeholders, such as customers, end-users, and business analysts.

**2. Design:** In this phase, the software design is created, which includes the overall architecture of the software, data structures, and interfaces. It has two steps:

* **High-level design (HLD):** It gives the architecture of software products.
* **Low-level design (LLD):** It describes how each and every feature in the product should work and every component.

**3. Implementation or coding:** The design is then implemented in code, usually in several iterations, and this phase is also called as Development.

things you need to know about this phase:

* This is the longest phase in SDLC model.
* This phase consists of Front end + Middleware + Back-end.
* **In front-end:**Development of coding is done even SEO settings are done.
* **In Middleware:** They connect both the front end and back end.
* **In the back-end:** A database is created.

**4. Testing:**The software is thoroughly tested to ensure that it meets the requirements and works correctly.

**5. Deployment:** After successful testing, The software is deployed to a production environment and made available to end-users.

**6. Maintenance:**This phase includes ongoing support, bug fixes, and updates to the software.

There are **different methodologies** that organizations can use to implement the SDLC, such as**Waterfall, Agile, Scrum, V-Model**and**DevOps.**

What is the Agile methodology?

The Agile methodology is a project management approach that involves breaking the project into phases and emphasizes continuous collaboration and improvement. Teams follow a cycle of planning, executing, and evaluating.

## What is Scrum?

Scrum is an agile development methodology used in the development of Software based on an iterative and incremental processes. Scrum is adaptable, fast, flexible and effective agile framework that is designed to deliver value to the customer throughout the development of the project.

|  |
| --- |
|  |

USE CASE

A use case is a concept used in software development, product design, and other fields to describe how a system can be used to achieve specific goals or tasks. It outlines the interactions between users or actors and the system to achieve a specific outcome.

### What is the purpose of a use case?

The purpose of a use case is to:

* Manage scope
* [Establish requirements](https://www.wrike.com/templates/requirements-management/)
* Outline the ways a user will interact with the system
* Visualize system architecture
* Communicate technical requirements to business stakeholders
* [Risk management](https://www.wrike.com/templates/project-risk-analysis/)

## How to write a use case for a project

* **System**: A system is the product, service, or software under discussion.
* **Actors**: An actor is a user or anything else that exhibits behaviour when interacting with the system. The actor could be another system, a piece of hardware, or an entire organization. There are four types of actors: a system under discussion, an internal actor, a primary actor, and a secondary actor. The most commonly referred to are the latter two systems. A primary actor initiates the interaction with the system, while a secondary actor may provide a service to the system.
* **Scenario**: In “Applying UML and Patterns,” Larman notes that “a scenario is a specific sequence of actions and interactions between actors and the system under discussion; it is also called a use case instance.”
* **Use case**: A use case outlines the success and failure scenarios that can occur when the actor(s) interact with the system. In this section, you’d establish the main success scenario, i.e., the most desirable outcome between the actor and the system. You would also establish the alternative paths, which explain what happens in the event of failure or error.

create a use case diagram for the "Facebook"

Use Case Diagram:

1. Actors:

- User: The main actor who interacts with the Facebook platform.

2. Use Cases:

* Create Account: This use case allows new users to create a new Facebook account.
  + Log In: This use case enables users to log in to their Facebook accounts.
  + Log Out: This use case allows users to log out from their Facebook accounts.
  + Post Status: This use case enables users to create and share a status update with their friends.
  + Upload Photo/Video: This use case allows users to upload photos or videos to their profile or a specific album.
  + Like Post: This use case allows users to like (show appreciation) for a post shared by others.
  + Comment on Post: This use case enables users to add comments to posts shared by others.
  + Share Post: This use case allows users to share a post from someone else's profile on their own timeline or with friends.
  + Send Message: This use case enables users to send private messages to other users or groups.
  + Add Friend: This use case allows users to send a friend request to connect with other users.
  + Accept Friend Request: This use case allows users to accept or reject friend requests received from others.
  + Search: This use case enables users to search for other users, groups, pages, or content within Facebook.
  + Manage Privacy Settings: This use case allows users to customize their privacy settings and control who can see their posts and profile information.

|  |
| --- |
| AGILE METHODOLOGY |

The Agile methodology is a project management approach that involves breaking the project into phases and emphasizes continuous collaboration and improvement. Teams follow a cycle of planning, executing, and evaluating.

DIFFERENT TYPES OF METHODOLOGY

* Scrum
* Crystal
* Kanban
* Dynamic Software Development Method(DSDM)
* Feature Driven Development(FDD)
* Lean Software Development
* eXtreme Programming(XP)

1.SCRUM :

Scrum is an agile development methodology used in the development of Software based on an iterative and incremental processes. Scrum is adaptable, fast, flexible and effective agile framework that is designed to deliver value to the customer throughout the development of the project.

5 Scrum phases :

1. Initiation

* The initiation phase of a Scrum framework is the period in which you create a vision for your project. This includes important identification points, such as noting who the stakeholders are for the project and assigning the role of Scrum Master to yourself or another member of the team responsible for executing the plan. This is also the period in which you assign team members to the project in order to create your team.

2. Planning and estimation

* During this phase, you create plans for a sprint, which is a short, time-boxed period that can help your team collaborate more effectively. As your team completes each sprint, you can then combine them later to complete all necessary elements in the project backlog. Consider selecting relevant items from the backlog and moving them into your sprint backlog when creating the plans.

3. Implementation

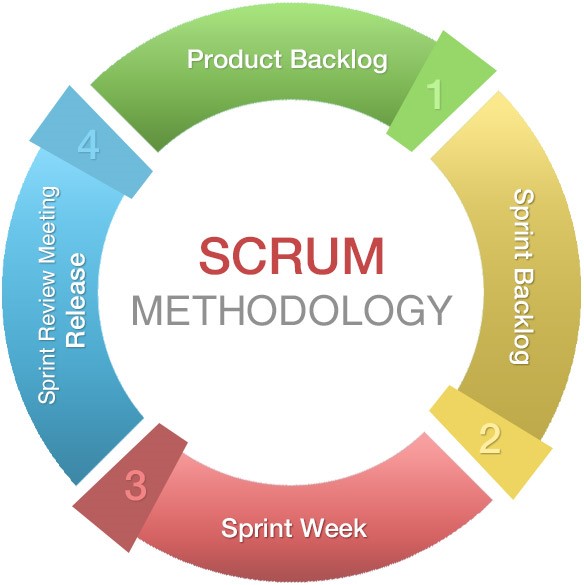
* The implementation phase is when you and your team implement the sprint as planned. During this phase, you maintain an updated backlog, removing items as staff complete them and assigning out new items from the backlog as needed. Consider meeting to provide project updates and review the work plans or concerns. During this meeting, encourage staff to ask questions, make requests or submit important notes that could be valuable for the other members to hear.

4. Reviewing

* To get feedback, consider scheduling a review meeting with your team at the end of the project to discuss the sprint. This meeting provides an opportunity to discuss what went well and where there are areas for improvement based on the results of the completed sprint. It allows you to adjust processes and procedures in order to be successful when transitioning into the next planning and estimation phase.

5. Releasing

* The last phase is the release phase, in which you deliver any final products to stakeholders, such as bringing a product to market or providing a client with the developed technology. After releasing the product, consider organizing a project retrospective meeting with your team to analyze the performance of each individual sprint and to discuss the overall performance of the project.



Advantages of Scrum:

* Flexibility
* Customer-centric
* Transparency
* Collaboration
* Continuous improvement
* Early and frequent delivery
* Reduced risk
* Empowered teams

Disadvantages of Scrum:

* Initial complexity
* Lack of predictability
* Time-boxed sprints
* Dependency on team collaboration
* Inadequate adaptation
* Dependency on a competent Product Owner
* Organizational resistance

2.CRYSTAL :

The crystal method is an agile framework that is considered a lightweight or agile methodology that focuses on individuals and their interactions. The methods are color-coded to significant risk to human life. It is mainly for short-term projects by a team of developers working out of a single workspace.

Steps in Crystal Agile Framework:

Project Kick-off and Team Formation:

* Assemble a small, cross-functional team consisting of individuals with the necessary skills and expertise to complete the project.

Communication and Collaboration:

* Emphasize open and frequent communication among team members, stakeholders, and customers to ensure a shared understanding of the project's goals and requirements.

Incremental Development:

* Adopt an incremental and iterative approach to software development, delivering working increments of the product regularly.

Prioritization and Planning:

* Collaborate with stakeholders and customers to prioritize features and requirements.
* Plan iterations (or increments) based on the prioritized items.

Continuous Feedback and Improvement:

* Regularly gather feedback from stakeholders and customers to validate the product's direction and make necessary adjustments.
* Hold regular retrospectives to identify areas for improvement in the development process.

Empowerment and Self-Organization:

* Encourage self-organization within the development team, allowing team members to make decisions about how best to achieve project goals.

Risk Management:

* Identify and manage project risks proactively to minimize their impact on the project.

Quality Focus:

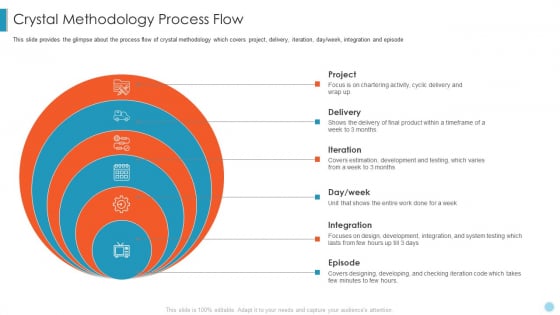
* Prioritize and maintain a strong focus on delivering a high-quality product.
* Use best practices for coding, testing, and deployment.

Deliver Working Software Regularly:

* Regularly deliver working increments of the product, ensuring that each increment adds value to the customer.

Adaptability and Evolution:

* Be open to adapting the development process as the project progresses and new information becomes available.



ADVANTAGES OF CRYSTAL :

* Flexibility
* Communication-focused
* Simplicity
* Continuous improvement
* Team morale
* Reduced overhead
* Suitable for small teams

DISADVANTAGES OF CRYSTAL :

* Lack of prescriptive guidance
* Limited suitability for large projects
* Dependency on team competency
* Difficulty in scaling
* Resistance to change
* Lack of standardization

3.KANBAN:

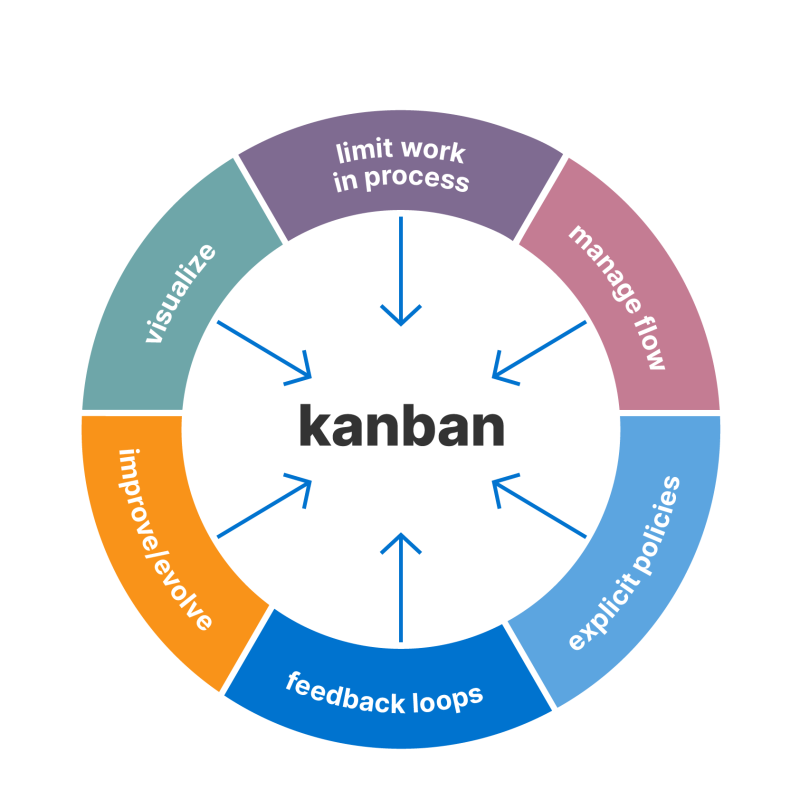
* Kanban is a popular project management and workflow methodology that originated in Japan and was later adopted in software development and various other industries. It focuses on visualizing work, limiting work in progress (WIP), and optimizing flow to improve productivity and efficiency. The word "Kanban" comes from two Japanese characters: "kan" meaning "visual," and "ban" meaning "card" or "board."

The Kanban methodology principles:

* Visualize the workflow: Represent the entire workflow visually using a Kanban board, which typically consists of columns representing different stages of the process and cards representing individual tasks or work items.
* Limit work in progress: Place WIP limits on each column to prevent overloading the team and to maintain a smooth flow of work.
* Manage flow: Focus on completing tasks in progress before starting new ones to optimize the flow and avoid bottlenecks.
* Make process policies explicit: Clearly define the rules and guidelines for moving tasks between columns to ensure everyone understands the workflow and follows the same process.
* Continuously improve: Regularly analyze the workflow, identify inefficiencies or areas for improvement, and make adjustments to enhance productivity.

steps to implementing a Kanban system:

* Visualize your current workflow.
* Apply Work-in-Process (WIP) limits.
* Make policies explicit.
* Manage and measure flow.
* Optimize iteratively with data.



ADVANTAGES OF KANBAN :

* Flexibility
* Improved workflow
* Reduced waste
* Enhanced collaboration
* Continuous improvement

DISADVANTAGES OF KANBAN :

* Lack of structure for some projects
* Less predictability
* Requires discipline
* Limited for new teams

1. Dynamic Software Development Method(DSDM)

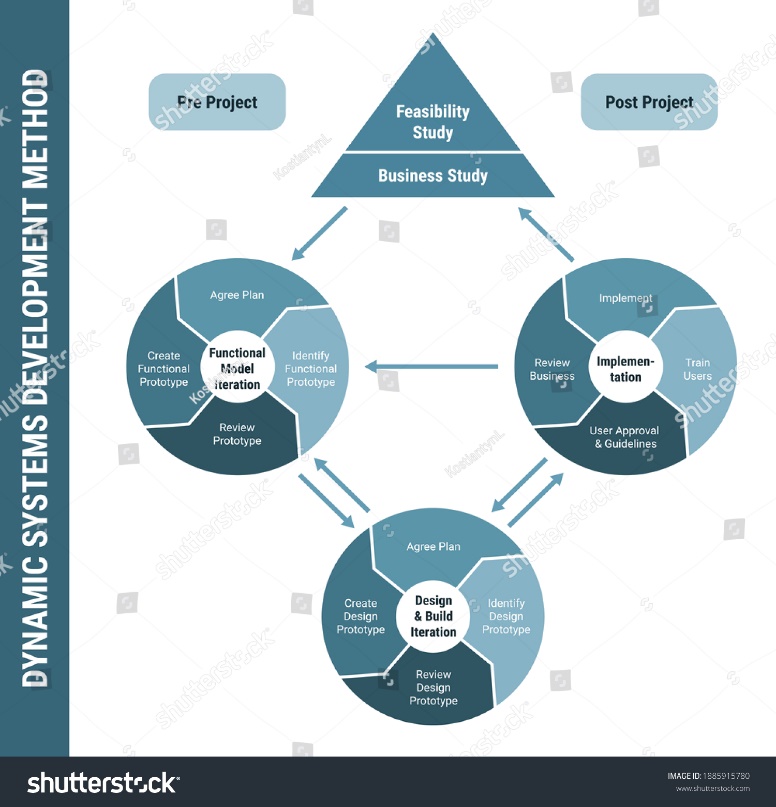
Dynamic Software Development Method (DSDM) is an agile project delivery framework designed to provide a structured and disciplined approach to software development. It focuses on delivering functional software quickly and frequently, ensuring active involvement of stakeholders throughout the development process

Principles Of DSDM:

* Focus on the Business Need
* Deliver on Time
* Collaborate
* Never Compromise on Quality
* Build Incrementally from Firm Foundations
* Develop Iteratively
* Communicate Continuously and Clearly
* Demonstrate Control

Phases of DSDM:

* Feasibility Study: In this initial phase, the project team identifies the business needs and constraints, assesses the feasibility of the project, and produces a high-level overview of the solution.
* Business Study: Here, the focus is on understanding the business requirements in detail. The team identifies stakeholders, gathers more specific requirements, and defines the project scope.
* Functional Model Iteration: In this phase, the team starts to build a functional model or a prototype of the solution. It allows stakeholders to visualize and interact with the proposed system, gathering early feedback.
* Design and Build Iteration: During this step, the team refines the functional model, adds more detail to the design, and starts building the solution in increments or iterations.
* Implementation: This phase involves integrating the developed increments into the existing system, conducting system testing, and addressing any issues that arise.
* Deployment: The final product is deployed and made available to end-users. The team ensures that proper training, documentation, and support are provided.
* Post-Project Review: After the solution is deployed, the team conducts a review to evaluate the project's success and learn from the process. This feedback is used to improve future projects.



ADVANTAGES OF DSDM:

* Agile and Flexible
* Reduced Time to Market
* Improved Quality
* Risk Mitigation
* Clear Roles and Responsibilities
* Efficient Resource Management
* Continuous Learning and Improvement
* Increased Customer Satisfaction

DISADVANTAGES OF DSDM:

* Not Suitable for All Projects
* Dependency on Active Stakeholder Involvement
* Complex for Large Projects
* Learning Curve for New Teams
* Challenges in Distributed Development Teams
* Documentation and Processes May Require Extra Effort

### Difference between Scrum and Kanban:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | | S. No. | Scrum | Kanban | | --- | --- | --- | | 1. | It defines the role of each member of the Scrum team. | There is no role assigned to individuals. | | 2. | It follows the iterative method. | It does not follow the iterative approach | | 3. | To solve a problem, it breaks it into small tasks and then processes it further. | It does not break a problem into sub-problems. | | 4. | It is a highly prescriptive approach. | It is not much prescriptive as compared to Scrum. | | 5. | There is no visualization process to perform tasks. | There is a visualization process to perform tasks. | | 6. | There are sprints that keep track of the progress of any project. | They use task cards to keep track of the progress of any project. | | 7. | It is processed in successive sprints to complete a task. | It is used to optimize the task to complete a project. | | 8. | It is not preferred when resources are limited. | It is preferred when tasks and resources are limited. | | 9. | Scrum Master is the problem solver in case of a problem. | All the members are allowed to pick a problem and solve it. | | 10. | The process does not get disturbed if a team member leaves in between a sprint. | The flow of work gets disturbed if a team member leaves in between. | | 11. | The velocity of the sprint is used to measure the production. | The time taken to finish the project is the measure of production. | | 12. | Estimation is crucial to Scrum because it places a strong emphasis on planning. | Estimation is not as important in Kanban as in scrum. | | 13. | In scrum, cross-functional teams are important to deal with the issues that may occur during software development. | In Kanban, specialized teams are important. | | 14. | Only one team owns a sprint backlog. | The sharing among multiple teams is possible with Kanban board. | | 15. | The scrum methodology is centered on the backlog. | The Kanban methodology is centered on the process dashboard. | | 16. | It is suitable for projects that have changing priorities. | It is suitable for projects that have stable priorities i.e. unlikely to change over time. | | 17. | “Velocity” through sprints is a production measurement metric. | “Cycle time” is a production measurement metric. | | 18. | One to four weeks make up a sprint cycle. | The delivery cycle is continuous. | | 19. | **Some of the Tools-**   * Jira Software * Axosoft * VivifyScrum and more. | **Some of the Tools-**   * Jira Software * Kanbanize * SwiftKanban | | |