**What is Manual testing?**

Manual Testing is a type of Software Testing where Testers manually execute test cases without using any automation tools. Manual Testing is the most primitive of all testing types and helps find bugs in the software system.

Any new application must be manually tested before its testing can be automated. Manual Testing requires more effort but is necessary to check automation feasibility.

### What is Software Testing?

Software testing is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is[Defect](https://www.guru99.com/the-unconventional-guide-to-defect-management.html)free. It involves execution of a software component or system component to evaluate one or more properties of interest.

Software testing also helps to identify errors, gaps or missing requirements in contrary to the actual requirements. It can be either done manually or using automated tools.

|  |  |
| --- | --- |
| **Manual Testing** | **Automated Testing** |
| Manual testing requires human intervention for test execution. | [Automation Testing](https://www.guru99.com/automation-testing.html) is use of tools to execute test cases |
| Manual testing will require skilled labour, long time & will imply high costs. | Automation Testing saves time, cost and manpower. Once recorded, it's easier to run an automated test suite |
| Any type of application can be tested manually, certain testing types like ad-hoc and monkey testing are more suited for manual execution. | Automated testing is recommended only for stable systems and is mostly used for [Regression Testing](https://www.guru99.com/regression-testing.html) |
| Manual testing can become repetitive and boring. | The boring part of executing same test cases time and again is handled by automation software in Automation Testing. |

**Goal of Manual Testing**

The key concept of manual testing is to ensure that the application is error free and it is working in conformance to the specified functional requirements.

Test Suites or cases, are designed during the testing phase and should have 100% test coverage.

It also makes sure that reported defects are fixed by developers and re-testing has been performed by testers on the fixed defects.

Basically, this testing checks the quality of the system and delivers bug-free product to the customer.

## How to perform Manual Testing

1. Read and understand the software project documentation/guides. Also, study the Application Under Test (AUT) if available.
2. Draft Test cases that cover all the requirements mentioned in the documentation.
3. Review and baseline the test cases with Team Lead, Client (as applicable)
4. Execute the test cases on the AUT
5. Report bugs.
6. Once bugs are fixed, again execute the failing test cases to verify they pass.

## What is Agile Testing?

Unlike the WaterFall method, Agile Testing can begin at the start of the project with continuous integration between development and testing. Agile Testing is not sequential (in the sense it's executed only after coding phase) but continuous.

An agile team works as a single team towards a common objective of achieving Quality. Agile Testing has shorter time frames called iterations (say from 1 to 4 weeks). This methodology is also called release, or delivery driven approach since it gives a better prediction on the workable products in short duration of time.

### What is Software Testing Life Cycle (STLC)?

Software Testing Life Cycle (STLC) is defined as a sequence of activities conducted to perform Software Testing.

Below are the phases of STLC:

* Requirement Analysis
* Test Planning
* Test case development
* Test Environment setup
* Test Execution
* Test Cycle closure

## Requirement Analysis

During this phase, test team studies the requirements from a testing point of view to identify the testable requirements.

The QA team may interact with various stakeholders (Client, Business Analyst, Technical Leads, System Architects etc) to understand the requirements in detail.

Requirements could be either Functional (defining what the software must do) or Non Functional (defining system performance /security availability )

Automation feasibility for the given testing project is also done in this stage.

**Activities**

* Identify types of tests to be performed.
* Gather details about testing priorities and focus.
* Prepare [Requirement Traceability Matrix (RTM)](https://www.guru99.com/traceability-matrix.html).
* Identify test environment details where testing is supposed to be carried out.
* Automation feasibility analysis (if required).

**Deliverables**

* RTM
* Automation feasibility report. (if applicable)

## Test Planning

Typically, in this stage, a Senior QA manager will determine effort and cost estimates for the project and would prepare and finalize the Test Plan. In this phase, Test Strategy is also determined.

**Activities**

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

**Deliverables**

* [Test plan](https://www.guru99.com/test-plan.html) /strategy document.
* [Effort estimation](https://www.guru99.com/testing-estimation.html) document.

## Test Case Development

This phase involves the creation, verification and rework of test cases & test scripts. [Test data](https://www.guru99.com/software-testing-test-data.html), is identified/created and is reviewed and then reworked as well.

**Activities**

* Create test cases, automation scripts (if applicable)
* Review and baseline test cases and scripts
* Create test data (If Test Environment is available)

**Deliverables**

* Test cases/scripts
* Test data

## Test Environment Setup

Test environment decides the software and hardware conditions under which a work product is tested. Test environment set-up is one of the critical aspects of testing process and ***can be done in parallel with Test Case Development Stage***. ***Test team may not be involved in this activity*** if the customer/development team provides the test environment in which case the test team is required to do a readiness check (smoke testing) of the given environment.

**Activities**

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

**Deliverables**

* Environment ready with test data set up
* Smoke Test Results.

## Test Execution

During this phase, the testers will carry out the testing based on the test plans and the test cases prepared. Bugs will be reported back to the development team for correction and retesting will be performed.

**Activities**

* Execute tests as per plan
* Document test results, and log defects for failed cases
* Map defects to test cases in RTM
* Retest the[Defect](https://www.guru99.com/the-unconventional-guide-to-defect-management.html)fixes
* Track the defects to closure

**Deliverables**

* Completed RTM with the execution status
* Test cases updated with results
* Defect reports

## Test Cycle Closure

Testing team will meet, discuss and analyze testing artifacts to identify strategies that have to be implemented in the future, taking lessons from the current test cycle. The idea is to remove the process bottlenecks for future test cycles and share best practices for any similar projects in the future.

**Activities**

* Evaluate cycle completion criteria based on Time, Test coverage, Cost,Software, Critical Business Objectives, Quality
* Prepare test metrics based on the above parameters.
* Document the learning out of the project
* Prepare Test closure report
* Qualitative and quantitative reporting of quality of the work product to the customer.
* Test result analysis to find out the defect distribution by type and severity.

**Deliverables**

* Test Closure report
* Test metrics

**Functional testing types include:**

* Unit testing
* Integration testing
* System testing
* Sanity testing
* Smoke testing
* Interface testing
* Regression testing
* Beta/Acceptance testing

#### #1) Alpha Testing

It is the most common type of testing used in the Software industry. The objective of this testing is to identify all possible issues or defects before releasing it into the market or to the user.

Alpha testing is carried out at the end of the software development phase but before the Beta Testing. Still, minor design changes may be made as a result of such testing. [Alpha testing](https://www.softwaretestinghelp.com/what-is-alpha-testing-beta-testing/) is conducted at the developer’s site. In-house virtual user environment can be created for this type of testing.

#### #2) Acceptance Testing

An [acceptance test](https://www.softwaretestinghelp.com/what-is-acceptance-testing/) is performed by the client and verifies whether the end to end the flow of the system is as per the business requirements or not and if it is as per the needs of the end user. Client accepts the software only when all the features and functionalities work as expected.

It is the last phase of the testing, after which the software goes into production. This is also called User Acceptance Testing (UAT).

#### #3) Ad-hoc Testing

The name itself suggests that this testing is performed on [an ad-hoc](https://www.softwaretestinghelp.com/ad-hoc-testing/) basis i.e. with no reference to the test case and also without any plan or documentation in place for such type of testing. The objective of this testing is to find the defects and break the application by executing any flow of the application or any random functionality.

Ad-hoc testing is an informal way of finding defects and can be performed by anyone in the project. It is difficult to identify defects without a test case but sometimes it is possible that defects found during ad-hoc testing might not have been identified using existing test cases.

#### #4) Accessibility Testing

The aim of [accessibility testing](https://www.softwaretestinghelp.com/what-is-web-accessibility-testing/) is to determine whether the software or application is accessible for disabled people or not. Here disability means deaf, color blind, mentally disabled, blind, old age and other disabled groups. Various checks are performed such as font size for visually disabled, color and contrast for color blindness etc.

#### #5) Beta Testing

[Beta Testing](https://www.softwaretestinghelp.com/beta-testing/) is a formal type of software testing which is carried out by the customer. It is performed in **the Real Environment**before releasing the product to the market for the actual end users.

Beta testing is carried out to ensure that there are no major failures in the software or product and it satisfies the business requirements from an end-user perspective. Beta testing is successful when the customer accepts the software.

Usually, this testing is typically done by end-users or others. It is the final testing done before releasing an application for commercial purpose. Usually, the Beta version of the software or product released is limited to a certain number of users in a specific area.

So end user actually uses the software and shares the feedback to the company. Company then takes necessary action before releasing the software to the worldwide.

#### #6) Back-end Testing

Whenever an input or data is entered on front-end application, it stores in the database and the testing of such database is known as Database Testing or Backend testing. There are different databases like SQL Server, MySQL, and Oracle etc. Database testing involves testing of table structure, schema, stored procedure, data structure and so on.

In back-end testing GUI is not involved, testers are directly connected to the database with proper access and testers can easily verify data by running a few queries on the database. There can be issues identified like data loss, deadlock, data corruption etc during this back-end testing and these issues are critical to fixing before the system goes live into the production environment

#### #7) Browser Compatibility Testing

It is a subtype of Compatibility Testing (which is explained below) and is performed by the testing team.

[Browser Compatibility Testing](https://www.softwaretestinghelp.com/how-is-cross-browser-testing-performed/) is performed for web applications and it ensures that the software can run with the combination of different browser and operating system. This type of testing also validates whether web application runs on all versions of all browsers or not.

#### #8) Backward Compatibility Testing

It is a type of testing which validates whether the newly developed software or updated software works well with older version of the environment or not.

Backward Compatibility Testing checks whether the new version of the software works properly with file format created by older version of the software; it also works well with data tables, data files, data structure created by older version of that software. If any of the software is updated then it should work well on top of the previous version of that software.

#### #9) Black Box Testing

Internal system design is not considered in this type of testing. Tests are based on the requirements and functionality.

Detailed information about the advantages, disadvantages, and [types of Black box testing](https://www.softwaretestinghelp.com/black-box-testing/) can be seen here.

#### #10) Boundary Value Testing

This type of testing checks the behavior of the application at the boundary level.

[Boundary value Testing](https://www.softwaretestinghelp.com/what-is-boundary-value-analysis-and-equivalence-partitioning/) is performed for checking if defects exist at boundary values. Boundary value testing is used for testing a different range of numbers. There is an upper and lower boundary for each range and testing is performed on these boundary values.

If testing requires a test range of numbers from 1 to 500 then Boundary Value Testing is performed on values at 0, 1, 2, 499, 500 and 501.

#### #11) Branch Testing

It is a type of white box testing and is carried out during unit testing. Branch Testing, the name itself suggests that the code is tested thoroughly by traversing at every branch.

#### #12) Comparison Testing

Comparison of a product’s strength and weaknesses with its previous versions or other similar products is termed as Comparison Testing.

#### #13) Compatibility Testing

It is a testing type in which it validates how software behaves and runs in a different environment, web servers, hardware, and network environment. [Compatibility testing](https://www.softwaretestinghelp.com/software-compatibility-testing/) ensures that software can run on a different configuration, different database, different browsers, and their versions. Compatibility testing is performed by the testing team.

#### #14) Component Testing

It is mostly performed by developers after the completion of unit testing. [Component Testing](https://www.softwaretestinghelp.com/what-is-component-testing-or-module-testing/) involves testing of multiple functionalities as a single code and its objective is to identify if any defect exists after connecting those multiple functionalities with each other.

#### #15) End-to-End Testing

Similar to system testing, [End-to-end testing](https://www.softwaretestinghelp.com/what-is-end-to-end-testing/) involves testing of a complete application environment in a situation that mimics real-world use, such as interacting with a database, using network communications, or interacting with other hardware, applications, or systems if appropriate.

#### #16) Equivalence Partitioning

It is a testing technique and a type of Black Box Testing. During this [equivalence partitioning](https://www.softwaretestinghelp.com/what-is-boundary-value-analysis-and-equivalence-partitioning/), a set of group is selected and a few values or numbers are picked up for testing. It is understood that all values from that group generate the same output.

The aim of this testing is to remove redundant test cases within a specific group which generates the same output but not any defect.

Suppose, application accepts values between -10 to +10 so using equivalence partitioning the values picked up for testing are zero, one positive value, one negative value. So the Equivalence Partitioning for this testing is: -10 to -1, 0, and 1 to 10.

#### #17) Example Testing

It means real-time testing. Example testing includes the real-time scenario, it also involves the scenarios based on the experience of the testers.

#### #18) Exploratory Testing

Exploratory Testing is informal testing performed by the testing team. The objective of this testing is to explore the application and looking for defects that exist in the application. Sometimes it may happen that during this testing major defect discovered can even cause system failure.

During exploratory testing, it is advisable to keep a track of what flow you have tested and what activity you did before the start of the specific flow.

[An exploratory testing technique](https://www.softwaretestinghelp.com/what-is-exploratory-testing/) is performed without documentation and test cases.

#### #20) Functional Testing

This type of testing ignores the internal parts and focuses only on the output to check if it is as per the requirement or not. It is a Black-box type testing geared to the functional requirements of an application. For detailed information about Functional Testing click [here](https://www.softwaretestinghelp.com/guide-to-functional-testing/).

#### #21) Graphical User Interface (GUI) Testing

The objective of this GUI testing is to validate the GUI as per the business requirement. The expected GUI of the application is mentioned in the Detailed Design Document and GUI mockup screens.

The GUI testing includes the size of the buttons and input field present on the screen, alignment of all text, tables and content in the tables.

It also validates the menu of the application, after selecting different menu and menu items, it validates that the page does not fluctuate and the alignment remains same after hovering the mouse on the menu or sub-menu.

#### #22) Gorilla Testing

Gorilla Testing is a testing type performed by a tester and sometimes by developer the as well. In Gorilla Testing, one module or the functionality in the module is tested thoroughly and heavily. The objective of this testing is to check the robustness of the application.

#### #23) Happy Path Testing

The objective of Happy Path Testing is to test an application successfully on a positive flow. It does not look for negative or error conditions. The focus is only on the valid and positive inputs through which application generates the expected output.

#### #24) Incremental Integration Testing

[Incremental Integration Testing](https://www.softwaretestinghelp.com/incremental-testing/) is a Bottom-up approach for testing i.e continuous testing of an application when a new functionality is added. Application functionality and modules should be independent enough to test separately. This is done by programmers or by testers.

#### #25) Install/Uninstall Testing

[Installation and uninstallation testing](https://www.softwaretestinghelp.com/software-installationuninstallation-testing/) is done on full, partial, or upgrade install/uninstall processes on different operating systems under different hardware or software environment.

#### #26) Integration Testing

Testing of all integrated modules to verify the combined functionality after integration is [termed as Integration Testing](https://www.softwaretestinghelp.com/what-is-integration-testing/). Modules are typically code modules, individual applications, client and server applications on a network, etc. This type of testing is especially relevant to client/server and distributed systems.

#### #27) Load Testing

It is a type of non-functional testing and the objective of Load testing is to check how much of load or maximum workload a system can handle without any performance degradation.

[Load testing helps](https://www.softwaretestinghelp.com/introduction-to-performance-testing-loadrunner-training-tutorial-part-1/) to find the maximum capacity of the system under specific load and any issues that cause the software performance degradation. Load testing is performed using tools like[JMeter](https://www.softwaretestinghelp.com/jmeter-tutorials/), LoadRunner, WebLoad, Silk performer etc.

#### #28) Monkey Testing

[Monkey testing](https://www.softwaretestinghelp.com/what-is-monkey-testing-in-software-testing/) is carried out by a tester assuming that if the monkey uses the application then how random input, values will be entered by the Monkey without any knowledge or understanding of the application.

The objective of Monkey Testing is to check if an application or system gets crashed by providing random input values/data. Monkey Testing is performed randomly and no test cases are scripted and it is not necessary to

Monkey Testing is performed randomly and no test cases are scripted and it is not necessary to be aware of the full functionality of the system.

#### #29) Mutation Testing

[Mutation Testing](https://www.softwaretestinghelp.com/what-is-mutation-testing/) is a type of white box testing in which the source code of one of the program is changed and verifies whether the existing test cases can identify these defects in the system. The change in the program source code is very minimal so that it does not impact the entire application, only the specific area having the impact and the related test cases should able to identify those errors in the system.

#### #30) Negative Testing

Testers having the mindset of “attitude to break” and using negative testing they validate that if system or application breaks. [A negative testing technique](https://www.softwaretestinghelp.com/what-is-negative-testing/) is performed using incorrect data, invalid data or input. It validates that if the system throws an error of invalid input and behaves as expected.

#### #31) Non-Functional Testing

It is a type of testing for which every organization having a separate team which usually called as Non-Functional Test (NFT) team or Performance team.

[Non-functional testing involves](https://www.softwaretestinghelp.com/what-is-non-functional-testing/) testing of non-functional requirements such as Load Testing, Stress Testing, Security, Volume, Recovery Testing etc. The objective of NFT testing is to ensure whether the response time of software or application is quick enough as per the business requirement.

It should not take much time to load any page or system and should sustain during peak load.

#### #32) Performance Testing

This term is often used interchangeably with ‘stress’ and ‘load’ testing. [Performance Testing](https://www.softwaretestinghelp.com/introduction-to-performance-testing-loadrunner-training-tutorial-part-1/) is done to check whether the system meets the performance requirements. Different performance and load tools are used to do this testing.

#### #33) Recovery Testing

It is a type of testing which validates that how well the application or system recovers from crashes or disasters.

Recovery testing determines if the system is able to continue the operation after a disaster. Assume that application is receiving data through the network cable and suddenly that network cable has been unplugged.

Sometime later, plug the network cable; then the system should start receiving data from where it lost the connection due to network cable unplugged.

#### #34) Regression Testing

Testing an application as a whole for the modification in any module or functionality is termed as Regression Testing. It is difficult to cover all the system in [Regression Testing](https://www.softwaretestinghelp.com/regression-testing-tools-and-methods/), so typically [automation testing tools](https://www.softwaretestinghelp.com/automation-testing-tutorial-1/) are used for these types of testing.

#### #35) Risk-Based Testing (RBT)

In [Risk Based Testing](https://www.softwaretestinghelp.com/risk-management-during-test-planning-risk-based-testing/), the functionalities or requirements are tested based on their priority. Risk-based testing includes testing of highly critical functionality, which has the highest impact on business and in which the probability of failure is very high.

The priority decision is based on the business need, so once priority is set for all functionalities then high priority functionality or test cases are executed first followed by medium and then low priority functionalities.

The low priority functionality may be tested or not tested based on the available time.

The Risk-based testing is carried out if there is insufficient time available to test entire software and software needs to be implemented on time without any delay. This approach is followed only by the discussion and approval of the client and senior management of the organization.

#### #36) Sanity Testing

[Sanity Testing](https://www.softwaretestinghelp.com/smoke-testing-and-sanity-testing-difference/) is done to determine if a new software version is performing well enough to accept it for a major testing effort or not. If an application is crashing for the initial use then the system is not stable enough for further testing. Hence a build or an application is assigned to fix it.

#### #37) Security Testing

It is a type of testing performed by a special team of testers. A system can be penetrated by any hacking way.

[Security Testing](https://www.softwaretestinghelp.com/how-to-test-application-security-web-and-desktop-application-security-testing-techniques/) is done to check how the software or application or website is secure from internal and external threats. This testing includes how much software is secure from the malicious program, viruses and how secure and strong the authorization and authentication processes are.

It also checks how software behaves for any hackers attack and malicious programs and how software is maintained for data security after such a hacker attack.

#### #38) Smoke Testing

Whenever a new build is provided by the development team then the software testing team validates the build and ensures that no major issue exists.

The testing team ensures that the build is stable and a detailed level of testing is carried out further. [Smoke Testing](https://www.softwaretestinghelp.com/smoke-testing-and-sanity-testing-difference/) checks that no show stopper defect exists in the build which will prevent the testing team to test the application in detail.

If testers find that the major critical functionality is broken down at the initial stage itself then testing team can reject the build and inform accordingly to the development team. Smoke Testing is carried out to a detailed level of any functional or regression testing.

#### #39) Static Testing

Static Testing is a type of testing which is executed without any code. The execution is performed on the documentation during the testing phase. It involves reviews, walkthrough, and inspection of the deliverables of the project. Static testing does not execute the code instead of the code syntax, naming conventions are checked.

The [static testing](https://www.softwaretestinghelp.com/static-testing-and-dynamic-testing-difference/) is also applicable for test cases, test plan, design document. It is necessary to perform static testing by the testing team as the defects identified during this type of testing are cost-effective from the project perspective.

#### #40) Stress Testing

This testing is done when a system is stressed beyond its specifications in order to check how and when it fails. This is performed under heavy load like putting large number beyond storage capacity, complex database queries, continuous input to the system or database load.

#### #41) System Testing

Under [System Testing technique](https://www.softwaretestinghelp.com/system-testing/), the entire system is tested as per the requirements. It is a Black-box type testing that is based on overall requirement specifications and covers all the combined parts of a system.

#### #42) Unit Testing

Testing of an individual software component or module is termed as [Unit Testing](https://www.softwaretestinghelp.com/unit-testing/). It is typically done by the programmer and not by testers, as it requires a detailed knowledge of the internal program design and code. It may also require developing test driver modules or test harnesses.

#### #43) Usability Testing

Under [Usability Testing](https://www.softwaretestinghelp.com/usability-testing-guide/), User-friendliness check is done. Application flow is tested to know if a new user can understand the application easily or not, Proper help documented if a user gets stuck at any point. Basically, system navigation is checked in this testing.

#### #44) Vulnerability Testing

The testing which involves identifying of weakness in the software, hardware and the network is known as Vulnerability Testing. Malicious programs, the hacker can take control of the system, if it is vulnerable to such kind of attacks, viruses, and worms.

So it is necessary to check if those systems undergo Vulnerability Testing before production. It may identify critical defects, flaws in the security.

#### #45) Volume Testing

[Volume testing](https://www.softwaretestinghelp.com/what-is-volume-testing/) is a type of non-functional testing performed by the performance testing team.

The software or application undergoes a huge amount of data and Volume Testing checks the system behavior and response time of the application when the system came across such a high volume of data. This high volume of data may impact the system’s performance and speed of the processing time.

#### #46) White Box Testing

[White Box testing](https://www.softwaretestinghelp.com/white-box-testing-techniques-with-example/) is based on the knowledge about the internal logic of an application’s code.

It is also known as Glass box Testing. Internal software and code working should be known for performing this type of testing. Under these tests are based on the coverage of code statements, branches, paths, conditions etc.

**Difference between Smoke & Sanity Software Testing:**

* Smoke testing is a wide approach where all areas of the software application are tested without getting into too deep. However, a sanity software testing is a narrow regression testing with a focus on one or a small set of areas of functionality of the software application.
* The test cases for smoke testing of the software can be either manual or automated. However, a sanity test is generally without test scripts or test cases.
* Smoke testing is done to ensure whether the main functions of the software application are working or not. During smoke testing of the software, we do not go into finer details. However, sanity testing is a cursory software testing type. It is done whenever a quick round of software testing can prove that the software application is functioning according to business / functional requirements.
* Smoke testing of the software application is done to check whether the build can be accepted for through software testing. Sanity testing of the software is to ensure whether the requirements are met or not.

|  |  |
| --- | --- |
| **Verification** | **Validation** |
| Are we building the system right? | Are we building the right system? |
| **Verification** is the process of evaluating products of a development phase to find out whether they meet the specified requirements. | **Validation** is the process of evaluating software at the end of the development process to determine whether software meets the customer expectations and requirements. |
| The objective of Verification is to make sure that the product being develop is as per the requirements and design specifications. | The objective of Validation is to make sure that the product actually meet up the user’s requirements, and check whether the specifications were correct in the first place. |
| Following activities are involved in **Verification**: Reviews, Meetings and Inspections. | Following activities are involved in **Validation**: Testing like black box testing, white box testing, gray box testing etc. |
| **Verification** is carried out by QA team to check whether implementation software is as per specification document or not. | **Validation** is carried out by testing team. |
| Execution of code is not comes under **Verification**. | Execution of code is comes under **Validation**. |
| **Verification** process explains whether the outputs are according to inputs or not. | **Validation** process describes whether the software is accepted by the user or not. |
| **Verification** is carried out before the Validation. | **Validation** activity is carried out just after the Verification. |
| Following items are evaluated during **Verification**: Plans, Requirement Specifications, Design Specifications, Code, Test Cases etc, | Following item is evaluated during **Validation**: Actual product or Software under test. |
| Cost of errors caught in **Verification** is less than errors found in Validation. | Cost of errors caught in **Validation** is more than errors found in Verification. |
| It is basically manually checking the of documents and files like requirement specifications etc. | It is basically checking of developed program based on the requirement specifications documents & files. |

# Difference Between Test Plan and Test Strategy

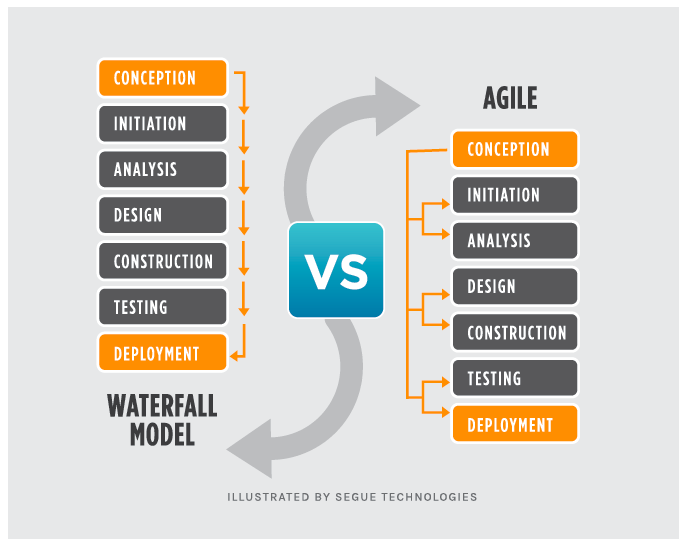
A Test Plan is a formal document derived from requirement documents, describing in detail the scope of testing and the different activities performed in testing.  
Whereas, a test strategy is a high-level document describing the way testing will be carried out in an organization.

|  |  |  |
| --- | --- | --- |
| **#** | **Test Plan** | **Test Strategy** |
| 1. | A test plan is derived from Software Requirement Specification (SRS), describing in detail the scope of testing and the different activities performed in testing. | A test strategy is a high-level document describing the way testing is carried out. |
| 2. | A test plan is specific to a particular project. | A test stategy is normally for a complete organization. Although it can be specifed for a particular project as well. |
| 3. | It describes the whole testing activities in detail - the techniques used, schedule, resources etc. | It describes the high-level test design techniques to be used, environment specifications etc. |
| 4. | It is prepared by test lead or test manager. | It is generally prepared by the project manager. |

**Waterfall** is a structured **software** development**methodology**, and often times can be quite rigid, whereas the **Agile methodology** is known for its flexibility. ... One of the major differences between **Agile** and **Waterfall**development **methodology** is their individual approach towards quality and testing.

**Here are the top 10 differences between Agile and Waterfall Methodology:**

1. The software development process is divided into different phases in the Waterfall model while Agile methodology segregates the project development lifecycle into sprints
2. Waterfall is a structured software development methodology, and often times can be quite rigid, whereas the Agile methodology is known for its flexibility
3. According to the Waterfall model, software development is to be completed as one single project, which is then divided into different phases, with each phase appearing only once during the SDLC. However, the Agile methodology can be considered as a collection of many different projects, which are nothing but the iterations of the different phases focusing on improving the overall software quality with feedbacks from users or the QA team
4. If you want to use the Waterfall model for software development, then you have to be clear with all the development requirements beforehand as there is no scope of changing the requirements once the project development starts. The Agile methodology, on the other hand, is quite flexible, and allows for changes to be made in the project development requirements even after the initial planning has been completed
5. All the project development phases such as designing, development, testing, etc. are completed once in the Waterfall model while as part of the Agile methodology, they follow an iterative development approach. As a result, planning, development, prototyping and other software development phases can appear more than once during the entire SDLC
6. One of the major differences between Agile and Waterfall development methodology is their individual approach towards quality and testing. In the Waterfall model, the “Testing” phase comes after the “Build” phase, but, in the Agile methodology, testing is typically performed concurrently with programming or at least in the same iteration as programming
7. While Waterfall methodology is an internal process and does not require the participation of customers, the Agile software development approach focuses on customer satisfaction and thus, involves the participation of customers throughout the development phase
8. The Waterfall model can be regarded as a stringently sequential process, however, the Agile methodology is a highly collaborative software development process, thereby leading to better team input and faster problem solving
9. The Waterfall model is best suited for projects which have clearly defined requirements and in which change is not expected at all, while Agile development supports a process in which the requirements are expected to change and evolve. Thus, if you are planning to develop a software that would require frequent overhauls and has to keep up with the technology landscape and customer requirements, Agile is the best approach to follow
10. The Waterfall model exhibits a project mindset and lays its focus strictly on the completion of project development, while Agile introduces a product mindset that focuses on ensuring that the developed product satisfies its end customers, and changes itself as the requisites of customers change



## What Is Agile Methodology?

Agile methodology is a practice that helps continuous iteration of development and testing in the SDLC process. Agile breaks the product into smaller builds.

In this methodology, development and testing activities are concurrent, unlike other software development methodologies. It also encourages teamwork and face-to-face communication. Business, stakeholders, and developers and clients must work together to develop a product.

## What Is Scrum?

Scrum is an agile process that allows us to focus on delivering the business value in the shortest time. It rapidly and repeatedly inspects actual working software. It emphasizes accountability, teamwork, and iterative progress toward a well-defined goal.

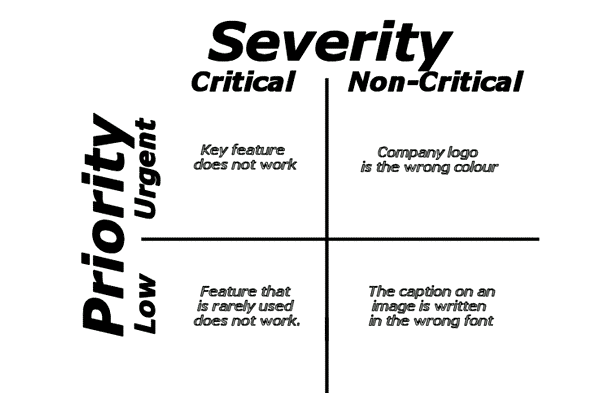
The Scrum Framework usually deals with the fact that the requirements are likely to change or most of the time not known at the start of the project.

## Agile Vs. Scrum

|  |  |
| --- | --- |
| **Agile** | **Scrum** |
| Agile is a development methodology based on **iterative and incremental approach**. | Scrum is one of the implementations of agile methodology. In which **incremental builds**are delivered to the customer in every two to three weeks' time. |
| Agile software development has been widely seen as highly suitedto environments which have **small but expert project** development team | Scrum is ideally used in the project where the requirement is **rapidly** changing. |
| In the Agile process, the **leadership**plays a vital role. | Scrum fosters a **self-organizing**, cross-functional team. |
| The biggest advantage of agile is its **flexibility**as it quickly reacts to changes. | Compared to Agile it is a more **rigid**method. So there is not much room for frequent changes. |
| Agile involves **collaborations** and face-to-face interactions between the members of various cross-functional teams. | In Scrum, collaboration is achieved in **daily stand up meeting** with a fixed role assigned to scrum master, product owner, and team members. |
| Agile can require l**ots of up-front development process** and organizational change. | Not **too many changes** needed while implementing scrum process. |
| The agile method **needs frequent delivery**to the end user for their feedback. | In the scrum, after each sprint, a **build is delivered** to the client for their feedback. |
| In this method, each step of development like requirements, analysis, design, are **continually monitored**during the lifecycle. | A demonstration of the functionality is provided at the end of every sprint. So that **regular feedback** can be taken before next sprint. |
| **Project head** takes cares of all the tasks in the agile method. | There is no team leader, so the **entire** **team addresses the issues** or problems. |
| The Agile method **encourages feedback**during the process from the end user. In this way, the end product will be more useful. | **Daily sprint meeting** is conducted to review and feedback to decide future progress of the project. |
| **Deliver and update**the software on a regular basis. | When the team is done with the **current sprint activities**, the next sprint can be planned. |
| Design and execution should be kept **simple**. | Design and execution can be **innovative and experimental**. |
| In the Agile method, the priority is always to satisfy the customer by providing **continuous delivery** of valuable software. | **Empirical Process Control** is a core philosophy of Scrum based process. |
| Working software is the most **elementary measure** of progress. | Working software is **not an elementary measure**. |
| It is best to have **face-to-face communication**, and techniques like these should be used to get as close to this goal as possible. | Scrum team focus to deliver **maximum business value**, from beginning early in the project and continuing throughout. |

**severity vs priority**

**Severity** is defined as the degree of impact a defect has on the development or operation of a component application being tested. **Priority** is defined as the order in which a defect should be fixed. Higher the **priority** the sooner the defect should be resolved.



**What is Severity?**  
Severity is defined as the degree of impact a defect has on the development or operation of a component application being tested.

Higher effect on the system functionality will lead to the assignment of higher severity to the bug. Quality Assurance engineer usually determines the severity level of defect  
\*\*  
What is Priority?\*\*  
Priority is defined as the order in which a defect should be fixed. Higher the priority the sooner the defect should be resolved.

Defects that leave the software system unusable are given higher priority over defects that cause a small functionality of the software to fail.

Defect severity can be categorized into four class

Critical: This defect indicates complete shut-down of the process, nothing can proceed further  
Major: It is a highly severe defect and collapse the system. However, certain parts of the system remain functional  
Medium: It cause some undesirable behavior, but the system is still functional  
Low: It won't cause any major break-down of the system

Defect priority can be categorized into three class

Low: The defect is an irritant but repair can be done once the more serious defect have been fixed  
Medium: During the normal course of the development activities defect should be resolved. It can wait until a new version is created  
High: The defect must be resolved as soon as possible as it affects the system severely and cannot be used until it is fixed

**Difference between Priority and Severity**:

**Priority:** Defect Priority is defined the order in which the developer should resolve a defect  
**Severity :** Defect Severity is defined as the degree of impact that a defect has on the operation of the product

**Priority** is categorized into three types  
Low  
Medium  
High

\**Severity \**are categorized into five types  
Critical  
Major  
Moderate  
Minor  
Cosmetic

**Priority:** is associated with scheduling  
**Severity :** is associated with functionality or standards

**Priority:** indicates how soon the bug should be fixed  
\**Severity: \**indicates the seriousness of the defect on the product functionality

\**Priority \**: of defects is decided in consultation with the manager/client  
\**Severity: \**QA engineer determines the severity level of the defect

**Priority** is driven by business value  
\**Severity \**is driven by functionality

**Priority** : Its value is subjective and can change over a period of time depending on the change in the project situation  
\**Severity \**: Its value is objective and less likely to change

**Priority**High priority and low severity status indicates, defect have to be fixed on immediate bases but does not affect the application  
\**Severity \**High severity and low priority status indicates defect have to be fixed but not on immediate bases

**Priority \**status is based on the customer requirements  
\**Severity** status is based on the technical aspect of the product

**Priority**During UAT the development team fix defects based on priority  
**Severity**During SIT, the development team will fix defects based on the severity and then priority

### What is JIRA?

JIRA is a tool developed by Australian Company Atlassian. It is used for **bug tracking, issue tracking,** and **project management**. The name "JIRA" is actually inherited from the Japanese word "Gojira" which means "Godzilla".

The basic use of this tool is to track issue and bugs related to your software and[Mobile](https://www.guru99.com/mobile-testing.html)apps. It is also used for project management. The JIRA dashboard consists of many useful functions and features which make handling of issues easy. Some of the key features are listed below. Let's learn JIRA[Defect](https://www.guru99.com/the-unconventional-guide-to-defect-management.html)and Project tracking software with this Training Course.

# Use Jira applications and Confluence together

confluence and Jira are like bacon and eggs; coffee and cake; Simon and Garfunkel. Separately, they're great, but [together, they're amazing](https://www.atlassian.com/software/confluence/jira-integration)!

If your Confluence and Jira sites are connected using [Application Links](https://confluence.atlassian.com/doc/linking-to-another-application-360677690.html), you can display and create Jira issues and more from within Confluence.

What you can do with Confluence and Jira depends on the Jira application and version you have. Find out about the [required applications and versions](https://confluence.atlassian.com/doc/use-jira-applications-and-confluence-together-427623543.html#UseJiraapplicationsandConfluencetogether-jirarequired) later in this page.

## For every project or team

### Display issues on a page

You can display Jira issues on a Confluence page using the Jira Issues macro. Display a single issue, a list of issues, or show the total number of issues.

The simplest way to add a Jira issue to Confluence is to paste a Jira URL on a Confluence page.

[Here's some examples...](https://confluence.atlassian.com/doc/use-jira-applications-and-confluence-together-427623543.html)

Alternatively, you can add the [Jira Issues Macro](https://confluence.atlassian.com/doc/jira-issues-macro-139380.html) to the page and search for issues directly:

1. In the editor choose **Insert** > **Jira Issue**.
2. Follow the prompts in the macro browser to choose a project and search for an issue – you can even use [JIRA Query Language (JQL)](https://confluence.atlassian.com/jirasoftwareserver/advanced-searching-939938733.html).

Once you've added the macro, you can customize how the issue or list of issues appears on the page, including how much information to display, how many issues, [and more](https://confluence.atlassian.com/doc/jira-issues-macro-139380.html).

### Create reports and charts

Reporting on information stored in Jira is simple in Confluence. In addition to the Jira Issues Macro, you can use the Jira Report blueprint or Jira Chart macro to show information from your Jira application visually. It's the best way to give your stakeholders a snapshot of your team or project's progress.

You can:

* Use the [JIRA Report blueprint](https://confluence.atlassian.com/doc/jira-report-blueprint-427623492.html) to create a Change Log or Status report.
* Use the [Jira Chart Macro](https://confluence.atlassian.com/doc/jira-chart-macro-427623467.html) to display data as a chart, including pie charts, created vs resolved, and two dimensional charts.
* Use JIRA [Gadgets](https://confluence.atlassian.com/doc/gadgets-204047542.html) to display detailed Jira reports and charts on pages.

### Create issues from inside Confluence

You can create issues while viewing a page or from the within the editor. This is really useful if you use Confluence for planning and gathering requirements.