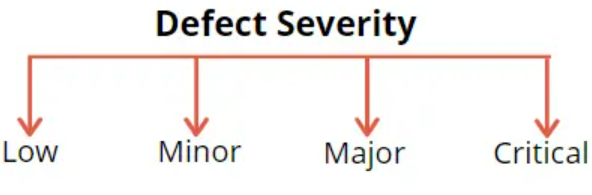
|  |
| --- |
| \Topics covered |
| * Defect Severity * Defect Priority * Severity vs Priority * Introduction to Software Testing * Software Testing Life Cycle * Different phases in STLC |

**Defect Severity**

* Defect Severity means how badly the defect has affected or impacted the application’s functionality.
* A higher effect of bug/defect on system functionality will lead to a higher severity level.

**Types of Severity**

Types of Severity of bug/defect can be categorized into the following parts:



**Critical**:

* A defect that has completely blocked the functionality of an application where the user or the tester cannot proceed or test anything.
* If the whole application’s functionality is inaccessible or down because of a defect, such a defect is categorized as a critical defect.
  + Example1 - When the login screen of an application is not working and the user cannot log in, the whole application becomes inaccessible to the user.
* Example2 – cannot login to an application, application hangs, crashes, closes abruptly after entering username and password, corrupts data in a database.

**Major**:

* It is a highly severe defect and collapses the system.
* When a bug doesn’t affect the entire application but still some significant system functionalities are not working, those defects can be considered as a Major defect.
* Example1 - In a banking application, a user cannot transfer money to any beneficiary account, but still add beneficiary is working as expected.

**Minor\Medium**:

* The behaviour of an application is different than expected, but this does not affect functionality.
* The minor defects will have a workaround, so it may not block a system\application functionality completely.
* These minor defects can wait until the next release because they do not restrict or block the application’s functionality.
* Example - The download link in the Help section of an application needs to be fixed. However, the user is still able to read the document online.

**Low**:

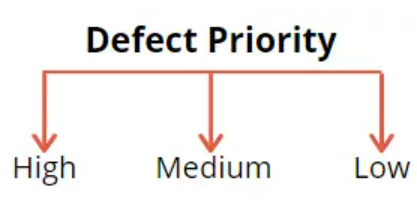
* Defects of cosmetic nature that don’t affect the application functionality. But they are valid defects and needs to be fixed.
* Example - Cosmetics or Spelling mistakes on the webpage. These are valid defects, but they can wait to be fixed since they’re not affecting application functionality.

**Defect Priority**

* **Defect Priority**defines the order in which developers will fix defects (because priority describes business importance).
* How quickly the defects need to be fixed.
* Higher the impact of a bug on a business’s functionality, the higher the priority assigned to the defect.

**Priority Types**

Types of Priority of bug/defect can be categorized into three parts :



**High:**

* If a defect directly affects the application, it is marked as a high priority.
* These bugs may affect the whole application and needs to be resolved as soon as possible.
* The application cannot be accessible until it is fixed.

**Medium:**

* The defects which don’t affect business and customers.
* They are less urgent than high-priority defects and can be fixed when the development team has the bandwidth to take them up.
* Such bugs can be fixed either in the same release or the next release.

**Low:**

* The defects that have the least priority for getting fixed are fixed after all the high and medium-priority defects are fixed.
* The fix for low-priority defects is usually provided along with some tall or medium-priority defect fixes.

**Severity vs Priority**

|  |  |
| --- | --- |
| **Priority** | **Severity** |
| Defect Priority has defined the order in which the developer should resolve a defect | Defect Severity is defined as the degree of impact that a defect has on the operation of the product |
| Priority indicates how soon the bug should be fixed | Severity indicates the impact of the defect on the product functionality |
| Priority is driven by business value | Severity is driven by functionality |
| Priority status is decided by on customer requirements | Severity status is based on the technical aspect of the product or the testers |
| Deals with the timeframe or order to fix the defects. | Deals with the technical aspects of the application. |
| During UAT the development team fix defects based on priority | During SIT, the development team will fix defects based on the severity and then priority |

## Severity and Priority Real-time Examples

The priority and severity are combined in four different ways to determine which defect needs immediate attention and which one the least.

* High Severity and High Priority - User unable to logon to the application
* Low Severity vs. High Priority - Company's logo is misspelled on the welcome page of the application.
* High Severity vs. Low Priority - The application is crashing on providing very large input for processing.
* Low Severity vs. Low Priority - The colour of any text does not match the theme of the application. If the privacy policy of the website has a spelling mistake

**Software Testing**

**Introduction to Software Testing**

* Software testing can be stated as the process of verifying and validating whether a software or application is bug-free.
* Meets the technical requirements as guided by its design and development.
* Meets the user requirements effectively and efficiently by handling all the exceptional and boundary cases.
* The purpose of software testing is to identify errors, gaps, or missing requirements in contrast to actual requirements.

**Software testing can be divided into two steps:**

**Verification**:

* **Verification** is the process of checking whether the software product has been developed without any error or bugs.
* It is the process to ensure whether the product that is developed is right or not.
* It verifies whether the developed product fulfils the requirements that we have.
* Verification means **Are we building the product, right?**

**Validation:**

* Validation is the process of checking the quality of the developed software product by performing different testing.
* It is validation of actual and expected product results.
* Validation means **Are we building the right product?**

## Why Software Testing is Important?

* Software Testing is Important because if there are any bugs or errors in the software, it can be identified at the earlier stage and can be resolved before the delivery of the software product.
* Properly tested software product ensures Quality, reliability, high performance which further results in time saving, cost and customer satisfaction.

**Software Testing Life Cycle (STLC)**

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

**Different phases in STLC**

There are following six major phases in every Software Testing Life Cycle Model (STLC Model):

**A diagram of software testing life cycle

Description automatically generated**

1. Requirement Analysis.
2. Test Planning.
3. Test case development.
4. Test Environment setup.
5. Test Execution.
6. Test Cycle closure.

**What is Entry and Exit Criteria in STLC?**

* **Entry Criteria:** Entry Criteria gives the prerequisite items that must be completed before testing.
* **Exit Criteria:** Exit Criteria defines the items that must be completed once after testing is completed.

**Requirement Analysis**

* Also known as Requirement Phase Testing
* Testing team analyse the requirements and interact with various stakeholders to understand requirements in detail.
* Requirements could be either functional or non-functional.

**Activities**

* Identify types of tests to be performed.
* Identify the test cases and test scenarios.
* Prepare [Requirement Traceability Matrix (RTM)](https://www.guru99.com/traceability-matrix.html).
* Identify test environment details where testing is supposed to be carried out.

**Deliverables**

* RTM

|  |
| --- |
| **Requirement Traceability Matrix (RTM)** is a document that maps and track user requirements with test cases.  It captures all requirements proposed by the client and requirement traceability in a single document.  The main purpose of RTM document is to validate that all requirements are checked via test cases such that no functionality is unchecked during Software testing.  **Parameters for RTM-**   * Requirement ID * Requirement Type and Description * Test Cases with Status   **A screenshot of a computer  Description automatically generated** |

## Test Planning in STLC

* Testmanager determines the test plan strategy along with efforts and cost estimates for the project.
* Resources, test environment, test limitations (In scope\Out of scope) and the testing schedule are also determined.
* The Test Plan gets prepared and finalized in the same phase.

**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/strategy document.
* [Effort estimation](https://www.guru99.com/an-expert-view-on-test-estimation.html) document.

## Test Case Development Phase

* Involves the creation of test cases & test scripts after the test plan is ready.
* [Test data](https://www.guru99.com/software-testing-test-data.html) is identified then created and reviewed.
* Testing team starts the development process of test cases for individual units.

**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/ test scripts.
* Test data.

**Test Environment Setup**

* Decides the software conditions under which a software product is tested.
* It is one of the critical aspects of the testing process and can be done in parallel with the Test Case Development Phase.
* Test team may not be involved in this activity if the development team provides the test environment.
* 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 Phase

* **Test Execution Phase** is carried out by the testers in which testing of the software build is done based on test plans and test cases prepared.
* The process consists of test script execution, test script maintenance and bug reporting.
* If bugs are reported, then it is reverted to development team for bug fixes 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/defect-management-process.html)fixes.
* Track the defects to closure.

**Deliverables**

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

**Test Cycle Closure**

* Test Cycle Closure phase is completion of test execution where testing team should be ready with the test artifacts like test completion report, test execution results.
* Testing team can discuss and analyse the lessons learnt during the entire testing process and it can be documented.
* The known issues and the expected errors during UAT can be documented and shared with customers.

**Activities**

* Prepare test execution results.
* Document the lessons learnt during the testing cycle.
* Prepare Test closure report.

**Deliverables**

* Test Closure report.

|  |
| --- |
| Topics covered |
| * Test Strategy * Test Plan * Source to Target Mapping Document * Test Case Document * Test Summary Document |

## Test Strategy in Software Testing

* A **Test Strategy** is a plan for defining a test approach to the Software Testing Life Cycle (STLC).
* It guides QA teams to define [Test Coverage](https://www.guru99.com/test-coverage-in-software-testing.html)and testing scope (In & Out scope).
* It helps testers get a clear picture of the project at any instance.
* If there is proper test strategy document in, there won’t be any miss in the test activity.

**Test Strategy Document**

* **Test Strategy Document** is a well-described document in software testing which clearly defines the exact software testing approach and testing objectives of the software application.
* It is an important document for QA teams which is derived from actual business requirements that will provide proper guidance to the whole team about software testing approach and objectives for each activity in the software testing process.
* A Test strategy document answers all the questions like what you want to get done and how you are going to accomplish it.
* Testing strategy plan should be communicated with the entire team so that the team will be consistent on approach and responsibilities.

**Parameters\Steps Involved in Test Strategy**

**Scope**

It defines parameters like

* Who will review the document?
* Who will approve this document?
* Software Testing activities carried out with timelines.

**Test Approach**

It defines parameters like

* Process of testing
* Testing levels
* Roles and responsibilities of each team member
* Types of Testing (Load testing, Security testing, Performance testing etc.)
* Testing approach & automation tool if applicable
* Adding new defects, re-testing, Defect triage, Regression Testing and test sign off.

**Test Environment**

It defines parameters like

* Define the number of requirement and setup required for each environment.
* Define backup of test data and restore strategy.

**Testing Tools**

It defines parameters like

* Automation and Test management tools needed for test execution.
* Figure out number of open source as well as commercial tools required and determine how many users are supported on it and plan accordingly.

**Release Control**

It defines parameters like

* Release management plan with appropriate version history that will make sure test execution for all modification in that release.

**Risk Analysis**

It defines parameters like

* List all risks that you can estimate.
* Give a clear plan to mitigate the risks also a contingency plan.

**Review and Approvals**

* All these activities are reviewed and signed off by the business team, project management etc.
* Summary of review changes should be traced at the beginning of the document along with an approved date, name, and comment.

**Components of Test Strategy Document**

**A diagram of components of a strategy document

Description automatically generated**

**Scope and Overview:**

* Defines the document’s approval, review, and usage.
* Specifies testing activities and phases needing approval.
* Overview of the project along with in scope and out scope details.

**Testing Methodology:**

* Details levels of testing, procedures, team roles, and responsibilities.
* Includes change management process for modifying requests.

**Testing Environment Specifications:**

* Specifies test data requirements and preparation guidelines.
* Outlines the number of environments and their setup requirements.
* Includes backup and restore strategies to prevent data loss.

**Testing Tools:**

* Details test management and automation tools for test execution.
* Defines tools for performance, and load testing, including tool types and user capacity.

**Release Control:**

* Ensures effective test execution and release management strategies.

**Risk Analysis:**

* Describes potential project risks impacting test execution.
* Includes strategies to mitigate risks and a contingency plan.

**Review and Approvals:**

* Involves review by Project Management, Development, and Business Teams.

**Test Plan**

* A [test plan](https://www.geeksforgeeks.org/software-testing-test-plan-estimates-and-strategy/) is a document that consists of all future testing-related activities.
* Prepared at the project level.
* Test manager will be preparing a test plan.
* The test plan serves as the blueprint that changes according to the progressions in the project and stay updated.
* It serves as a base for conducting testing activities and coordinating activities among a QA team.
* It is shared with Business Analysts, Project Managers, and other stakeholders involved.

**Why Test Plans are Important:**

The following are some of the key benefits of making a test plan:

* Quick guide for the testing process.
* Helps to avoid out-of-scope functionalities.
* Helps to determine the time, cost, and effort.
* Provide a schedule for testing activities.
* Test plan can be reused.

**Components and Attributes of Test Plan document**

## A group of white squares with numbers Description automatically generated

**Objective –**

* **Describes the overall objective of the test plan or the project.**
* **List all the functionality and performance to be tested.**
* **Make goals and targets based on the application feature.**

**Scope:**

**It consists of information that needs to be tested in an application. The scope can be divided into two parts:**

* **In-Scope: The modules or the applications or the database objects that are to be tested.**
* **Out Scope: The modules that are not to be tested.**

**Testing Methodology:**

* **The methods that are going to be used for testing.**
* **The testing methodology is decided based on the feature and application requirements.**
* **It determines What Type of testing will be used in the testing methodology.**
* **API testing, system testing, functional testing, non-functional testing, application testing, integration testing etc.,**

**Approach:**

* **It determines the test approach for the end-to-end testing flow of the application along with high-Level Scenarios.**

**Assumption**

* **The testing team is not responsible for the test data availability.**
* **The tester will get proper knowledge transfer from the development team.**
* **Testing will be performed only in testing environment.**
* **Only Performance baseline testing would be performed.**
* **Other assumptions related to business, or the functionality needs to be considered.**

**Risk: All the risks related to project and the testing must be listed down.**

* **Not ready with environment setup**
* **Delay in Test data availability.**
* **Delay in the status of development activity.**

**Mitigation Plan:**

* **There should be mitigation plan for the above risks mentioned, so that the project can be delivered on time without any delay.**

**Roles and Responsibilities:**

* **Roles and responsibility of the tester and the test manager should be determined.**

**Schedule –**

* **Timelines, basically the start and end time for test case design and test execution should be determined.**

**Defect Tracking –**

* **While performing the end-to-end testing, testers can identify the bugs or defects which must be reported to the development team for fix.**
* **All those defects must track and maintained in a defect management tool.**

**Test Environments**

* **Test environments used for end-to-end testing process.**
* **The software installation details.**

**Entry and Exit Criteria:**

* **The set of conditions that should be met to start any new type of testing or to end any kind of testing.**

**Test Automation:**

* **It consists of the features that are to be automated and which features are not to be automated.**
* **If there any test cases which are executed multiple times, those can be considered for automation.**
* **Regression testing scenarios can be automated.**

**Effort Estimation:**

* **This involves planning the efforts that needs to be applied by every team member throughout the testing cycle.**
* **Efforts for test design, test execution, retesting of the bugs, regression testing .**

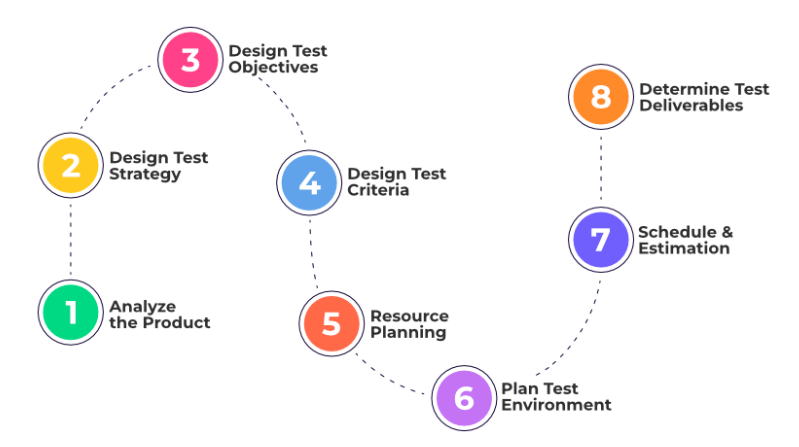
**Test Deliverables:**

* **Test artifacts like test execution report, defect report and RTM document has to be given to the customer at the end of the testing phase.**

**Template:**

* **The test plan template must be followed the testing team.**
* **All the test engineers will only use these templates in the project to maintain the consistency of the product.**

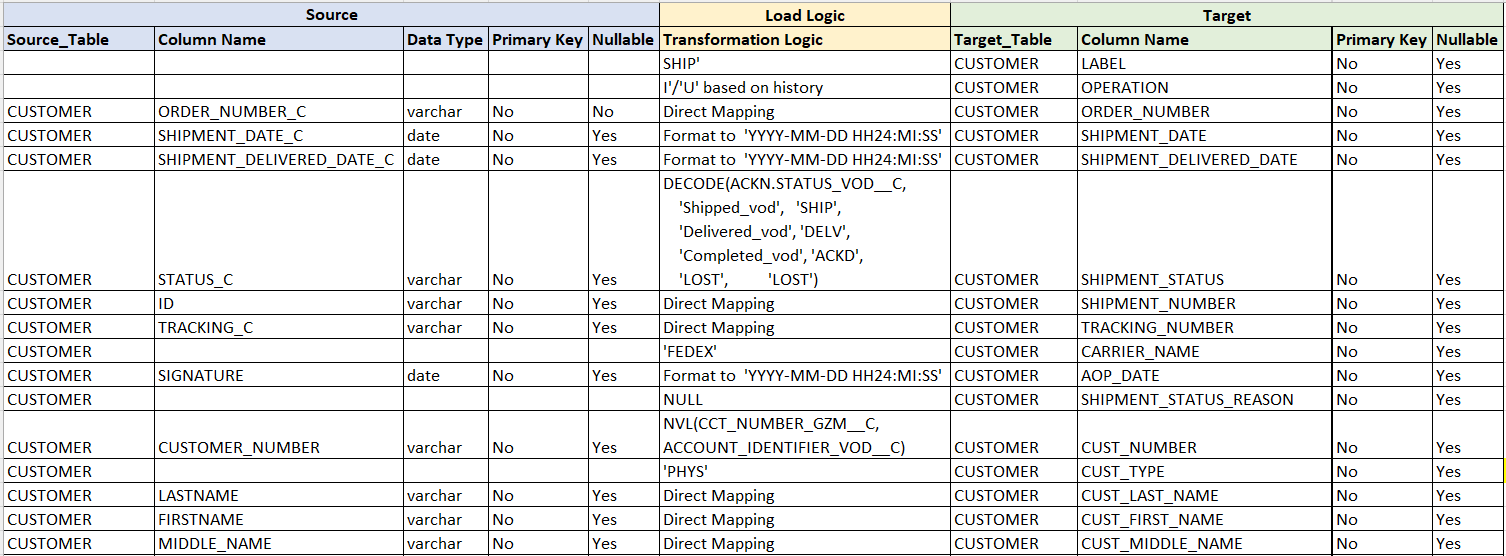
**How to create Test plan**

****

|  |  |
| --- | --- |
| **Difference Between Test Strategy and Test Plan** | |
| **Test Strategy** | **Test Plan** |
| Test strategy is a set of guidelines that explains test design and determines how testing needs to be done | A test plan for software project can be defined as a document that defines the scope, objective, approach. |
| Components of Test strategy includes- objectives and scope, documentation formats, test processes, team reporting structure, client communication strategy, etc. | Components of Test plan include- Test plan id, features to be tested, test techniques, testing tasks, features pass or fail criteria, test deliverables, responsibilities, and schedule, etc. |
| A test strategy is carried out by the project manager. It says what type of technique to follow and which module to test | Test plan is carried out by a testing manager or lead that describes how to test, when to test, who will test and what to test |
| Test strategy narrates about the general approaches | Test plan narrates about the specification |
| Test strategy cannot be changed | Test plan can change |
| It is a long-term plan of action. You can abstract information that is not project specific and put it into test approach | Test planning is done to determine possible issues and dependencies to identify the risks. |

**Source to Target Mapping Document**

* **Source-to-Target Mapping is a set of data transformation instructions that determine how to convert the structure and content of data in the source system to the structure and content needed in the target system.**
* **Source-to-Target Mapping solutions enable their users to identify columns or keys in the source system and point them to columns or keys in the target systems.**

****

**Test Case Document-**

* A Test Case document is a well-designed document for developing and better understanding of the test case for a particular test case scenario.

**Test case document will contain below parameter,**

* **Test case ID: Each test case should be represented by a unique ID.**
* **Test Designed by: Tester’s Name**
* **Date of test designed: Date when test was designed.**
* **Test Executed by: Who executed the test- tester.**
* **Date of the Test Execution: Date when test needs to be executed.**
* **Description/Summary of Test: Determine the summary or test purpose in brief.**
* **Pre-condition: Any requirement that needs to be done before execution of this test case. To execute this test case, list all pre-conditions.**
* **Test Steps: Mention all the test steps in detail and write in the order in which it requires to be executed.**
* **Expected Results: Mention the expected result including error or message that should appear on screen.**
* **Actual Result: After test execution, actual test result should be filled.**
* **Status (Fail/Pass): Mark this field as failed if actual result is not as per the estimated result.**

**Test Summary Document**

* **Test Report is a document which contains a summary of all test activities and final test results of a testing project.**

## What does a test report contain?

## Project Information

## Project name

## Description

**Test Objective**

* **Test Type**
* **Purpose**

**Test Summary**

* **Test cases passed.**
* **Test cases failed.**
* **Test cases blocked.**

**Defect**

* **Defect description.**
* **Priority**
* **Status**

**Project Information**

* All information of the project such as the project name, product name, and version should be described in the test report.

### **Test Objective**

* Test Report should include the objective of each round of testing, such as Unit Test, Performance Test, System Test …Etc.

### **Test Summary**

This section includes the summary of testing activity which includes,

* The number of test cases executed.
* The numbers of test cases pass.
* The numbers of test cases fail.
* Pass percentage.
* Fail percentage.

### **Defect**

One of the most important factors in Test Report is defect. The report should contain following information.

* Total number of bugs
* Status of bugs (open, closed, responding)
* Number of bugs open, resolved, closed.
* Breakdown by severity and priority