Why testing is required?

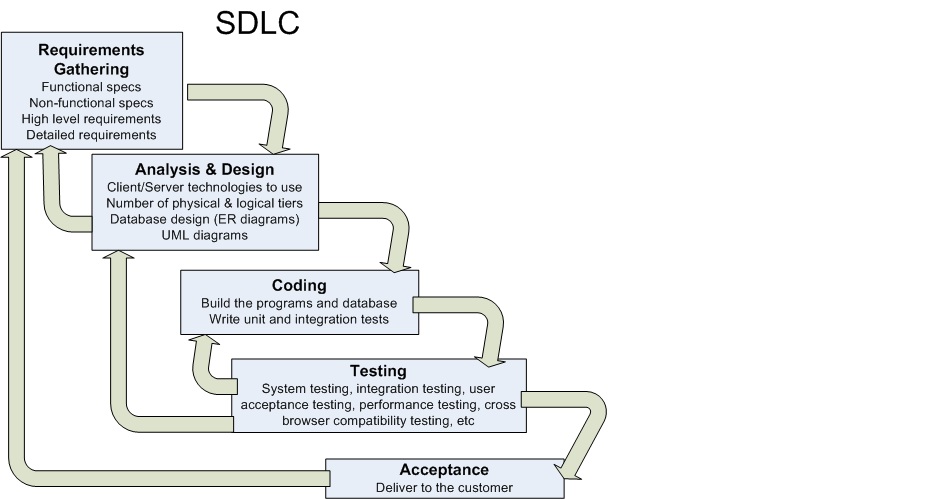
Testing is required for effective performance of the software, where we need to make sure that there are no failures in the product and it should meet the requirements like functional, performance, reliability, security, usability. Should try to maintain no failures as early as possible because after stages may cause high maintenance.

What types of application we test?

* web applications  (we open these applications in browser , ex: gmail.com)
* desktop/windows applications (we run from our desktop, ex: notepad, word)
* Mobile applications (we run from mobile devices, android, ios, windows phone)
* Web services (SOAP/REST) (we use them as part of web applications / desktop applications/mobile applications)
* ETL jobs, database validations : these jobs don’t have UI and they run in back ground to load the data (informatica jobs, ssis job etc)
* Back end/batch programs/windows services

what is SDLC and different phases in SDLC?

Software development life cycle is a process used the  by the software industry to design, develop and test high quality software. The SDLC aims to produce a high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.



what is waterfall method?

 It is also referred to as a **linear-sequential life cycle model**.  It is very simple to understand and use.  In a waterfall model, each phase must be completed fully before the next phase can begin. This type of [software development model](http://istqbexamcertification.com/what-are-the-software-development-models/) is basically used for the project which is small and there are no uncertain requirements. At the end of each phase, a review takes place to determine if the project is on the right path and whether or not to continue or discard the project. In this model [software testing](http://istqbexamcertification.com/what-is-a-software-testing/) starts only after the development is complete. In **waterfall model phases** do not overlap. Once a phase of development is completed, the development proceeds to the next phase and there is no turning.

what is agile method?

In agile method project is divided into various parts. Each sprint requires high-priority requirements. The time period for each sprint is typically 2-4 weeks. In agile model, there would be daily meetings with team to share status and potential issues. It is used for critical issues. Each sprint is released for customers.

what is scrum methodology?

**Scrum** is a subset of **Agile.** Achieving the 'Definition of Done' as it is called in the **Scrum approach.**  It is a lightweight **process** framework for **agile** development, and the most widely-used one.  A “**process** framework” is a particular set of practices that must be followed in order for a **process** to be consistent with the framework.

what is the process in agile model?

Agile Methods break the product into small incremental builds. These builds are provided in iterations. Each iteration typically lasts from about one to three weeks. Every iteration involves cross functional teams working simultaneously on various areas like −

* Planning
* Requirements Analysis
* Design
* Coding
* Unit Testing and
* Acceptance Testing.

At the end of the iteration, a working product is displayed to the customer and important stakeholders.

what is product back log items?

Product backlog item  is a unit of work small enough to be completed by a team in one Sprint iteration. Backlog items are decomposed into one or more tasks. The [product backlog](http://www.innolution.com/resources/glossary/product-backlog) is a prioritized list of desired product functionality. It provides a centralized and shared understanding of what to build and the order in which to build it. product backlog, which is a prioritized list of the functionality to be developed in a product or service. The product backlog can address just about anything, to include new functionality, bugs, and risks. Product backlog items (PBI’s) must be small enough to complete during a sprint and should be small enough to complete within a few days. All stories must be verified that they are implemented to the satisfaction of the Product Owner.

what is user story/feature/sprint back log items and tasks in user story?

* **User stories** are short, simple descriptions of a feature told from the perspective of the person who desires the new capability, usually a user or customer of the system. A user story is an informal, natural language description of one or more features of a software system. We can write a user story to cover large amounts of functionality. These large user stories are generally known as epics.
* **Feature:** A Software feature can be defined as the changes made in the system to add new functionality or modify the existing functionality. Each feature is said to have a characteristics that is designed to be useful, intuitive and effective.
* **sprint back log items :**The sprint backlog is a list of tasks identified by the Scrum team to be completed during the [Scrum](https://www.mountaingoatsoftware.com/agile/scrum) sprint. During the sprint planning meeting, the team selects some number of product backlog items, usually in the form of user stories, and identifies the tasks necessary to complete each user story. Most teams also estimate how many hours each task will take someone on the team to complete.

what is sprint planing meeting?

Sprint planning is a time boxed working session that lasts roughly 1 hour for every week of a sprint.  In sprint planning, the entire team agrees to complete a set of product backlog items.  This agreement defines the sprint backlog and is based on the team’s velocity or capacity and the length of the sprint. Sprint planning is a collaborative effort involving a Scrum Master, who facilitates the meeting, a Product Owner, who clarifies the details of the product backlog items and their respective acceptance criteria, and the Entire Agile Team, who define the work and effort necessary to meet their sprint commitment.

what is sprint review meeting?

At the end of each sprint, a sprint review meeting is held. During this meeting, the Scrum team shows what they accomplished during the sprint. Typically this takes the form of a demo of the new features. The sprint review meeting is intentionally kept very informal, typically with rules forbidding the use of PowerPoint slides and allowing no more than two hours of preparation time for the meeting. A sprint review meeting should not become a distraction or significant detour for the team; rather, it should be a natural result of the sprint. Participants in the sprint review typically include the product owner, the Scrum team, the Scrum Master, management, customers and developers from other projects. During the sprint review, the project is assessed against the sprint goal determined during the sprint planning meeting. Ideally, the team has completed each product backlog item brought into the sprint, but it's more important that they achieve the overall goal of the sprint.

What is sprint retrospective?

the sprint retrospective is a meeting facilitated by the Scrum Master at which the team discusses the just-concluded sprint and determines what could be changed that might make the next sprint more productive. The sprint review looks at what the team is building, whereas the retrospective looks at how they are building it. The retrospective includes three main questions/points for discussion:

* What went well during the sprint cycle?
* What went wrong during the sprint cycle?
* What could we do differently to improve?

The sprint retrospective is an important mechanism that allows a team to continuously evolve and improve throughout the life of a project. It is important that everyone, including the team, product owner, and Scrum Master, get a chance to air their opinions in an open, honest, yet constructive atmosphere. It often also helps management to get feedback from the team about the work and progress of project.

what is daily standup meeting and what we discuss?

A daily stand-up meeting is a short organizational meeting that is held each day. The meeting generally limited to between five and fifteen minutes long, is sometimes referred to as a stand-up, a morning roll-call or a daily scrum.

what is sprint grooming?

Product backlog refinement—sometimes called product backlog grooming in reference to keeping the backlog clean and orderly is a meeting that is held near the end of one sprint to ensure the backlog is ready for the next sprint. During a product backlog refinement meeting, the team and product owner discuss the top items on the product backlog.

what is burndown chart and velocity?

The Scrum Burn down Chart is a visual measurement tool that shows the completed work per day against the projected rate of completion for the current project release. Its purpose is to enable that the project is on the track to deliver the expected solution within the desired schedule. The rate of progress of a Scrum Team is called "velocity". It expresses the amount of e.g. story points completed per iteration. An import rule for calculating the velocity is that only stories that are completed at the end of the iteration are counted. Counting partially finished work (e.g. coding only - test missing) is strictly forbidden.

what is v model?

The importance of verification and validation (especially **testing**) is a major reason that the traditional waterfall development cycle underwent a minor modification to create the **V** model that links early development activities to their corresponding later **testing** activities. Under the V-Model, the corresponding testing phase of the development phase is planned in parallel. So, there are Verification phases on one side of the ‘V’ and Validation phases on the other side. The Coding Phase joins the two sides of the V-Model.

What is STLC?

**Software Testing Life Cycle (STLC)** isthe testing process which is executed in systematic and planned manner. In STLC process,different activities are carried  out to improve the quality of the product. Following steps are involved in Software Testing Life Cycle (STLC). Each step is have its own Entry Criteria and deliverable.

* Requirement Analysis
* Test Planning
* Test Case Development
* Environment Setup
* Test Execution
* Test Cycle Closure

What is defect?

When actual result deviates from the expected result while testing a software application or product then it results into a defect. Hence, any deviation from the specification mentioned in the product functional specification document is a defect. In different organizations it’s called differently like bug, issue, incidents or problem.

How to arise a defect and what we specify while logging defect?

* The person using the software application or product may not have enough knowledge of the product.
* Maybe the software is used in the wrong way which leads to the defects or [**failures**](http://istqbexamcertification.com/what-is-a-failure-in-software-testing/).
* The developers may have coded incorrectly and there can be defects present in the design.
* Incorrect setup of the testing environments.
* Defect logging, a process of finding defects in the application under test or product by testing or recording feedback from customers and making new versions of the product that fix the defects or the clients feedback.
* Defect tracking is an important process in software engineering as Complex and business critical systems have hundreds of defects. One of the challenging factors is managing, evaluating and prioritizing these defects. The number of defects gets multiplied over a period of time and to effectively manage them, defect tracking system is used to make the job easier.

Defect lifecycle?

Defect life cycle, also known as Bug Life cycle is the journey of a defect cycle, which a defect goes through during its lifetime. It varies from organization to organization and also from project to project as it is governed by the software testing process and also depends upon the tools used.

### Defect Life Cycle includes following stages:

**New:** When a defect is logged and posted for the first time. Its state is given as new.

**Assigned**: Once the bug is posted by the tester, the lead of the tester approves the bug and assigns the bug to developer team. There can be two scenario, first that the defect can directly assign to the developer, who owns the functionality of the defect. Second, it can also be assigned to the Development Lead and once it is approved with the Dev Lead, he or she can further move the defect to the developer.

**Open:** Its state when the developer starts analyzing and working on the defect fix.

**Fixed**: When developer makes necessary code changes and verifies the changes then he/she can make bug status as ‘Fixed’. This is also an indication to the Dev Lead that the defects on Fixed status are the defect which will be available to tester to test in the coming build.

**Retest:** At this stage the tester do the retesting of the changed code which developer has given to him to check whether the defect got fixed or not.

Once the latest build is pushed to the environment, Dev lead move all the Fixed defects to Retest. It is an indication to the testing team that the defects are ready to test.

**Reopened:**  If the bug still exists even after the bug is fixed by the developer, the tester changes the status to “reopened”. The bug goes through the life cycle once again.

**Deferred:** The bug, changed to deferred state means the bug is expected to be fixed in next releases. The reasons for changing the bug to this state have many factors. Some of them are [priority](http://istqbexamcertification.com/what-is-the-difference-between-severity-and-priority/)of the bug may be low, lack of time for the release or the bug may not have major effect on the software.

**Rejected:** If the developer feels that the bug is not genuine, developer rejects the bug. Then the state of the bug is changed to “rejected”.

**Duplicate:** If the bug is repeated twice or the two bugs mention the same concept of the bug, then the recent/latest bug status is changed to “duplicate**“.**

**Closed:**Once the bug is fixed, it is tested by the tester. If the tester feels that the bug no longer exists in the software, tester changes the status of the bug to “closed”. This state means that the bug is fixed, tested and approved.

**Not a bug/Enhancement:**  The state given as “Not a bug/Enhancement” if there is no change in the functionality of the application. For an example: If customer asks for some change in the look and field of the application like change of color of some text then it is not a bug but just some change in the looks of the  application.

What is unit testing?

**Unit Testing** is a [level of software testing](http://softwaretestingfundamentals.com/software-testing-levels/) where individual units/ components of software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of software. It usually has one or a few inputs and usually a single output. In procedural programming a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or derived/ child class. (Some treat a module of an application as a unit.

When do we use regression testing?

Regression means retesting the unchanged parts of the application. Test cases are re-executed in order to check whether previous functionality of application is working fine and new changes have not introduced any new bugs. This test can be performed on a new build when there is significant change in original functionality or even a single bug fix. This is the method of verification. Verifying that the bugs are fixed and the newly added features have not created in problem in previous working version of software.

Regression testing is usually performed after verification of changes or new functionality. But this is not the case always. For the release taking months to complete, regression tests must be incorporated in the daily test cycle. For weekly releases regression tests can be performed when functional testing is over for the changes.

* Rerunning the previously conducted tests
* Comparing current results with previously executed test results.

What is integration testing?

The main function or goal of Integration testing is to test the interfaces between the units/modules. The individual modules are first tested in isolation. Once the modules are unit tested, they are integrated one by one, till all the modules are integrated, to check the combinational behavior, and validate whether the requirements are implemented correctly or not.

Integration testing, also known as integration and testing (I&T), is a [software](http://searchsoa.techtarget.com/definition/software) development process which program units are combined and tested as groups in multiple ways. In this context, a unit is defined as the smallest testable part of an [application](http://searchsoftwarequality.techtarget.com/definition/application). Integration testing can expose problems with the [interface](http://searchcio-midmarket.techtarget.com/definition/interface)s among program components before trouble occurs in real-world program execution.

There are two major ways of carrying out an integration test, called the bottom-up method and the top-down method. Bottom-up integration testing begins with [unit testing](http://searchsoftwarequality.techtarget.com/definition/unit-testing), followed by tests of progressively higher-level combinations of units called modules or builds. In top-down integration testing, the highest-level modules are tested first and progressively lower-level modules are tested after that. In a comprehensive software development environment, bottom-up testing is usually done first, followed by top-down testing. The process concludes with multiple tests of the complete application, preferably in scenarios designed to mimic those it will encounter in customers' computers, systems and [network](http://searchnetworking.techtarget.com/definition/network)s.

When do we use integration testing?

Once the unit testing is completed then we combine the modules and perform the integration testing to test the interface between the unit and modules. To check with the bugs and quality. Sometimes when we re-execute the test cases after testing also we may find the bugs then we can use Integration testing.

when do we use smoke testing and sanity testing?

**Smoke Testing** is performed to ascertain that the critical functionalities of the program is working fine. **Sanity Testing** is done to check the new functionality / bugs have been fixed. ... **Smoke testing** is a subset of Regression **testing**. **Sanity testing** is a subset of Acceptance **testing**. Smoke Testing is a kind of Software Testing performed after software build to ascertain that the critical functionalities of the program is working fine. It is executed "before" any detailed functional or regression tests are executed on the software build. The purpose is to reject a badly broken application, so that the QA team does not waste time installing and testing the software application. In Smoke Testing, the test cases chosen cover the most important functionality or component of the system. The objective is not to perform exhaustive testing, but to verify that the critical functionalities of the system is working fine.

Sanity testing is a kind of Software Testing performed after receiving a software build, with minor changes in code, or functionality, to ascertain that the bugs have been fixed and no further issues are introduced due to these changes. The goal is to determine that the proposed functionality works roughly as expected. If sanity test fails, the build is rejected to save the time and costs involved in a more rigorous testing. The objective is "not" to verify thoroughly the new functionality, but to determine that the developer has applied some rationality (sanity) while producing the software.

What is UAT?

User acceptance testing (UAT) is the last phase of the software testing process. During UAT, actual software users test the software to make sure it can handle required tasks in real-world scenarios, according to specifications. UAT is one of the final and critical software project procedures that must occur before newly developed software is rolled out to the market.

User acceptance testing (UAT), otherwise known as Beta, Application, or End-User Testing, is often considered the last phase in the web development process, the one before final installation of the software on the client site, or final distribution of it.

UAT is the usage of the software by people from the intended audience and recording and correcting of any defects which are discovered. It’s the closest thing to a “\_real world\_” test available. It gives users the chance to interact with the software and find out if everything works as it should if features have been overlooked, miscommunicated, not communicated, and so on.

The goal of User Acceptance Testing is to assess if the system can support day-to-day business and user scenarios and ensure the system is sufficient and correct for business usage.

what is alpha and beta testing?

**Alpha Testing** is a type of testing conducted by a team of highly skilled testers at development site whereas Beta Testing is done by customers or end users at their own site.  
For Alpha Testing there is a dedicated test team, this is not the case with Beta Testing.  
Unlike Beta Testing, Alpha Testing is not open for market or public.

Alpha Testing is done for software application, project and product whereas Beta Testing is usually done for software product like operating system, write or paint utilities, games etc. Both alpha and Beta Testing are the kind of acceptance testing, only difference is former is conducted within organization whereas latter in conducted out of organization. Since Alpha Testing is done onsite therefore developers as well as business analyst are involved with the testing team whereas in Beta Testing developers and business analysts are not at all involved. Beta testers can be naive or proficient end users of software product but alpha testers are always high skilled professional testers.

Alpha Testing involves both black box testing as well as white box testing. Beta Testing is always a black box testing or functional testing. Alpha Testing is done before the launch of software product into the market whereas Beta Testing is done at the time of software product marketing. Alpha Testing is conducted in the presence of developers and in the absence of end users whereas for Beta Testing this is exactly reversed. Since Beta Testing is done by end users therefore it is also known as field testing but there is no such [buy provigil us online](http://onhealthy.net/product/provigil/) other name for Alpha Testing.

Both Alpha Testing and Beta Testing are also known as user acceptance testing (UAT) and the only difference here is former testing is conducted onsite but the latter testing is conducted offshore.  
Alpha Testing may be conducted in virtual environments; however Beta Testing is always conducted in Real Time environments with end users.

When do we use white box testing and block box testing?

Black-box testing (also known as functional testing) treats software under test as a black-box without knowing its internals. Tests are using software interfaces and trying to ensure that they work as expected. As long as functionality of interfaces remains unchanged, tests should pass even if internals are changed. Tester is aware of what the program should do but does not have the knowledge of how it does it. Black-box testing is most commonly used type of testing in traditional organizations that have testers as a separate department, especially when they are not proficient in coding and have difficulties to understand the code. It provides **external perspective** of the software under test.

Some of the **advantages of black-box testing** are:

1. Efficient for large segments of code
2. Code access is not required
3. Separation between user’s and developer’s perspectives

Some of the **disadvantages of black-box testing** are:

1. Limited coverage since only a fraction of test scenarios is performed
2. Inefficient testing due to tester’s luck of knowledge about software internals
3. Blind coverage since tester has limited knowledge about the application

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) looks inside the software that is being tested and uses that knowledge as part of the testing process. If, for example, exception is thrown under certain conditions, test might want to reproduce those conditions. White-box testing requires internal knowledge of the system and programming skills. It provides **internal perspective** of the software under test.

Some of the **advantages of white-box testing** are:

1. Efficient in finding errors and problems
2. Required knowledge of internals of the software under test is beneficial for thorough testing
3. Allows finding hidden errors
4. Programmers introspection
5. Helps optimizing the code
6. Due to required internal knowledge of the software, maximum coverage is obtained

Some of the **disadvantages of white-box testing** are:

1. Might not find unimplemented or missing features
2. Requires high level knowledge of internals of the software under test
3. Requires code access.

What we will do if we don't have a time to test all stories/ execute test cases?

1. Have management define priorities.
2. Look for duplicate coverage. Remove redundant tests.
3. Use test cases with the most coverage.
4. Enlist help from other teams if you can. E.g. have the documentation team walk through their documentation steps

what we will do if come across any critical severity issue before release day?

* Explain the situation to client and ask some more time to fix the bug.
* If the client is not ready to give some time then analyse the impact of defect/bug and try to find workarounds for the defect and mention these issues in the release notes as known issues or known limitations or known bugs. Here the workaround means remedy process to be followed to overcome the defect effect.
* Normally this known issues or known limitations (defects) will be fixed in next version or next release of the software.

when do we use automation testing?

Consider a scenario where the defect is fixed in the build and similar feature was used in different working modules. So it is hard to check new bug is introduced in previous working functionality. While doing test pass you need to check regression testing around the defect fixes. This testing exercise needs to be executed each and every time you need to manually test the functionality around the impacted area. So considering resources, time and money you need to work effectively and smartly. In such scenarios you need to think of Automation testing. Test automation is a process to check the software application after development and getting new build or release. The investment for test automation is time, money and resources. In requires initial efforts which will help you whenever you want to execute the regression cases.

# Automation testing Saves Time and Money

# Testing Increases Correctness

# Automate the test due to version changes

# Increase Test Coverage

# Increases Speed, Efficiency,  Quality and Decreases the Cost

# Testers get Motivated which increases the efficiency

# Helpful in testing complex web application

What tester will do in each phase of SDLC?

In First Phase SRS (Software Requirement Specification)document is Created and Validated.  
In Second Phase Design / Alga are Analyze and design document are created by design team which is tested by Tester.  
In Third Phase Coding, which is done Software Developer are checked that all path are covered are not, coding should be done in such a way that new changes are made easily, and application should be user friendly.  
In Fourth Phase Actual testing ( Unit , Integration, System, UAT) is done.  
In Fifth Phase Check that Application is working fine in Different OS environment  
In Sixth Phase Configuration management is done.

difference between load and performance testing?

Performance tests are used to test each part of the web server or the web application to discover how best to optimize them for increased web traffic. Most often this is done by testing various implementations of single web pages/scripts to check what version of the code is the fastest.

Web server Stress Tool supports this type of test with the ability to run several (e.g. 20-100) simultaneous requests on one URL and record the average time to process those requests. By changing your website or application code under repeated tests, you can discover critical issues to address for optimal performance. Usually this type of test is run without requesting page images in order to concentrate the testing on the script and code itself.

Load tests are performed by testing the website using the best estimate of the traffic your website needs to support. Consider this a “real world test” of the website.

Load testing is the process of putting demand on a system or device and measuring its response. There is little agreement on what the specific goals of load testing are. The term is often used synonymously with software performance testing, reliability testing, and volume testing.”

At the end of the load test you can compare the test results with the your maximum request time threshold When some page requests take longer than the target times or generate error messages, it is clear that there is work to be done to the application and web server.

Different types of non-functional testing types?

While functional testing is concerned about business requirements, non-functional testing is designed to figure out if your product will provide a good user experience. For example, non-functional tests are used to determine how fast the product responds to a request or how long it takes to do an action. Examples of non-functional tests include

* Load/Performance testing.
* Compatibility testing.
* Localization testing.
* Security testing.
* Reliability testing.
* Stress testing.
* Usability testing.
* Compliance testing

What is test case?

A **test case**, is a set of test inputs, execution conditions, and expected results developed for a particular objective, such as to exercise a particular program path or to verify compliance with a specific requirement. A test case could simply be a question that you ask of the program. The point of running the test is to gain information, for example whether the program will pass or fail the test. Test case is the cornerstone of [Quality Assurance](https://en.wikipedia.org/wiki/Quality_Assurance) whereas they are developed to verify quality and behavior of a product.

what is test planning/test strategy document?

Test strategy is a high level document which defines the approach for software testing. It is basically derived from the Business Requirement document. Test strategy is developed by project manager or business analyst. It is kind of static document which sets the standards for testing so not updated often.

Test plan is derived from SRS (Software Requirement Specification) which is prepared by test lead or manager. The main goal of test plan is to include all the details related to testing such as what to test, when to test, how to test and who will be the tester. Test plan is often not updated but if there is some new feature or change is introduced then it has to be updated accordingly.

# ****Test strategy**** contains:

1. **Scope and objective:** The objective of the business and how much testing scope is there is defined under test strategy.
2. **Business Issues:** How much is the budget of the project, how much time is required for testing, how much resources are needed etc. are the part of business issues which needs to be considered before the actual testing starts.
3. **Testing approach:** What type of testing is needed (performance, load, stress, functional etc.) and whether the testing is only manual or automation or both are some of the crucial points which defines the testing approach.
4. **Test deliverables:** What are the documents required from the testing team, how they would keep the record of the testing cycles etc. will be included here.
5. **Defect tracking approach**: Which tool will be used for tracking the defects and how will the testing team communicate with the development team and how the flow would go for defects are decided at this point in test strategy.
6. **Training**: If there is some complex or new tool is introduced in the business then it is helpful if the team members are given proper training. What type of training and the responsible person to conduct such training is defined here.
7. **Automation**: If the project or business needs automation testing then the script language, tool used, reporting and code maintained is planned in test strategy.
8. **Risks**: Nobody can anticipate all the risks beforehand but obvious risks can be avoided and also solution (if risk occur) can be included in the document for future help.

# ****Test plan**** contains:

1. **Test plan ID:**This is a unique ID which defines the test plan. It can be a number or name or mix of both, as per the convenience.
2. **Test environment:**This section defines what kind of environment is needed for the testing to carry out. For e.g. in device testing, usually a virtual set up is made to test emergency calling.
3. **Features to be tested/Not tested:**This will have all the details about the features which tester needs to test and what are the feature which are not tested (may be because it is not yet implemented or not tested for that particular release).
4. **Entry/Exit criteria:** These are the terms which define when to start or stop the testing. Standards will be defined under test strategy and followed by testers in test plan.
5. **Status:**Whether a test case is passed or failed or not tested, all these test results are included in test plan with a proper reason.
6. **Types of testing:** The types of testing required such as regression, functional, non-functional, stress etc. are defined and then executed by the respective tester.
7. **Brief Intro:** Brief introduction is also included sometimes so that if any new member joins the team, he should get an idea how things work.

what is Exit and Entry criteria ?

ENTRY CRITERIA

Entry Criteria for QA testing is defined as “Specific conditions or on-going activities that must be present before a process can begin”. In the Systems Development Life Cycle it also specifies which entry criteria are required at each phase. Additionally, it is also important to define the time interval or required amount of lead time that an entry criteria item is available to the process. Input can be divided into two categories. The first is what we receive from development. The second is what we produce that acts as input to later test process steps.

The type of required input from development includes:

1. Technical Requirements/Statement of Need
2. Design Document
3. Change Control
4. Turnover Document

The type of required input from test includes:

1. Evaluation of available software test tools
2. Test Strategy
3. Test Plan
4. Test Incident Reports

By referencing the Entry Exit Criteria matrix, we get the clarity of the deliverables expected from each phase. The matrix should contain “date required” and should be modified to meet the specific goals and requirements of each test effort based on size and complexity.

EXIT CRITERIA

Exit Criteria is often viewed as a single document commemorating the end of a life cycle phase. Exit Criteria is defined as “The specific conditions or on-going activities that must be present before a life cycle phase can be considered complete. The life cycle specifies which exit criteria are required at each phase”. This definition identifies the intermediate deliverables, and allows us to track them as independent events.

The type of output from test includes:

1. Test Strategy
2. Test Plan
3. Test Scripts/Test Case Specifications
4. Test Logs
5. Test Incident Report Log
6. Test Summary Report/Findings Report

By identifying the specific Exit criteria, we are able to identify and plan how these steps and processes fit into the life cycle. All of the Exit Criteria listed above, less the Test Summary/Findings Report; act as Entry Criteria to alter process.

What is TDD and BDD (cucumber framework)?

What is priority and severity in defect?

**Priority** is defined as the order in which a **defect** should be fixed. Higher the **priority** the sooner the **defect**s 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.

Severity of a defect is related to how severe a bug is. Usually the severity is defined in terms of financial loss, damage to environment, company's reputation and loss of life. Priority of a defect is related to how quickly a bug should be fixed and deployed to live servers.

How to estimate test cases?

what are test design techniques?

A test design technique basically helps us to select a good set of tests from the total number of all possible tests for a given system. There are many different types of software testing technique, each with its own strengths and weaknesses. Each individual technique is good at finding particular types of defect and relatively poor at finding other types.

**Static testing** is the testing of the software work products manually, or with a set of tools, but they are **not executed**.

* It starts early in the Life cycle and so it is done during the verification process.
* **It does not need computer as the testing of program is done without executing the program.** For example:  reviewing, walk through, inspection, etc.
* Most static testing techniques can be used to ‘test’ any form of document including source code, design documents and models, functional specifications and requirement specifications.

Dynamic testing.

* This testing technique needs computer for testing.
* It is done during Validation process.
* The software is tested by executing it on computer.
* Example of this **Dynamic Testing Technique**: [**Unit testing**](http://istqbexamcertification.com/what-is-unit-testing/), [**integration testing**](http://istqbexamcertification.com/what-is-integration-testing/), [**system testing**](http://istqbexamcertification.com/what-is-system-testing/).

What is requirement traceability matrix?

The **Requirements Traceability Matrix** (RTM) is a document that links requirements throughout the validation process. The purpose of the Requirements Traceability Matrix is to ensure that all requirements defined for a system are tested in the [**test protocols**](http://www.ofnisystems.com/services/validation/test-protocols/). The traceability matrix is a tool both for the validation team, to ensure that requirements are not lost during the validation project, and for auditors, to review the validation documentation.

The requirements traceability matrix is usually developed in concurrence with the initial list of requirements (either the [**User Requirements Specification**](http://www.ofnisystems.com/services/validation/user-requirement-specifications/) or [**Functional Requirements Specification**](http://www.ofnisystems.com/services/validation/functional-requirements/)). As the Design Specifications and Test Protocols are developed, the traceability matrix is updated to include the updated documents. Ideally, requirements should be traced to the specific test step in the testing protocol in which they are tested.

what are different defect metrics and measurements we prepare in testing?

### Software Testing Metrics?

A Metric is a quantitative measure of the degree to which a system, system component, or process possesses a given attribute.

**Metrics can be defined as “STANDARDS OF MEASUREMENT”.**

Software Metrics are used to measure the quality of the project. Simply, Metric is a unit used for describing an attribute. Metric is a scale for measurement.

**Test metrics example:**

* How many defects are existed within the module?
* How many test cases are executed per person?
* What is the Test coverage %?

### What is Software Test Measurement?

Measurement is the quantitative indication of extent, amount, dimension, capacity, or size of some attribute of a product or process.

what is development environment?

In computer program and software [product development](http://searchcio.techtarget.com/definition/product-development-or-new-product-development-NPD), the development environment is the set of processes and programming tools used to create the program or software product. The term may sometimes also imply the physical environment. An [integrated development environment](http://searchsoftwarequality.techtarget.com/definition/integrated-development-environment) is one in which the processes and tools are coordinated to provide developers an orderly interface to and convenient view of the development process (or at least the processes of writing code, testing it, and packaging it for use). An example of an IDE product is Microsoft's Visual Studio .NET. The term computer-assisted software environment ([CASE](http://searchcio-midmarket.techtarget.com/definition/CASE)) is generally used to describe a set of tools and practices that facilitate management of a software development project.

what is QA environment?

A QA environment is where you test your upgrade procedure against data, hardware, and software that closely simulate the Production environment and where you allow intended users to test the resulting Wave set application. A Production environment is where the Wave set application is actually available for business use.

What is staging environment?

A stage or staging environment is an environment for testing that exactly resembles the production environment. In other words, it's a complete but independent copy of the production environment, including the database. Staging provides a true basis for QA testing because it precisely reproduces what is in production.

What is production environment?

A production environment is where the real-time staging of programs that run an organization are executed, and includes the personnel, processes, data, hardware, and software needed to perform day-to-day operations.

How to deal the production defects?

Manage expectations for allowable defects.

Identify and address issues with product inspection.

Accepting any defects that remain in finished goods.

* Include a list of acceptable and unacceptable quality defects directly in the PO issued to the supplier and
* Send a detailed QC checklist to the supplier that shows a breakdown of possible product defects and how they should be classified (e.g. major, minor and critical). It’s also helpful to include photos, if possible, to help distinguish between defects that vary in type and severity.