**What is System Testing?**

In the world of software testing, system testing is the testing of a complete and fully integrated software product. It's The process of testing an integrated system to verify that it meets specified  
requirements. In main focus of this testing is to evaluate Business / Functional / End-user requirements.

**Purpose:**

The purpose of this test is to evaluate the system’s compliance with the specified requirements.

**ANALOGY**

During the process of manufacturing a ballpoint pen, the cap, the body, the tail, the ink cartridge and the ballpoint are produced separately and unit tested separately. When two or more units are ready, they are assembled and Integration Testing is performed. When the complete pen is integrated, System Testing is performed.

**Method:**

Black box testing method is used. This is type of testing where external working of the software is evaluated with the help of requirement documents & it is totally based on Users point of view. For this type of testing do not required knowledge of internal design or structure or code.

**When is it performed?**

This testing is to be carried out only after *System Integration Testing* is completed where both [Functional & Non-Functional requirements](http://www.softwaretestingclass.com/) are verified and before the acceptance testing.

In the integration testing testers are concentrated on finding bugs/defects on integrated modules. But in the ***Software System Testing*** testers are concentrated on finding bugs/defects based on software application behavior, software design and expectation of end user.

**Who performs it?**

Normally, independent Testers perform System Testing.

**Why system testing is important:**

a) In Software Development Life Cycle the System Testing is perform as the first level of testing where the System is tested as a whole.

b) In this step of testing check if system meets functional requirement or not.

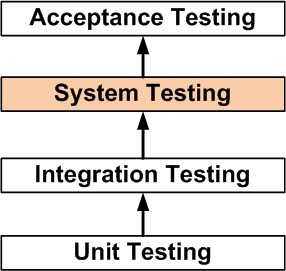
c) *System Testing* enables you to test, validate and verify both the Application Architecture and Business requirements.

d) The application/System is tested in an environment that particularly resembles the effective production environment where the application/software will be lastly deployed.

Generally, a separate and dedicated team is responsible for system testing. And System Testing is performed on staging server which is similar to production server. So this means you are testing software application as good as production environment.

**Different Hierarchical levels of testing:**

As with almost any technical process, software testing has a prescribed order in which things should be done. Different levels of testing are used in the testing process; each level of testing aims to test different aspects of the system. The following is lists of software testing categories arranged in sequentially organize.



**Unit testing** – Testing is done in the development process while developer completes the unit development.  The object of this testing is to verify correctness of the module. The purpose of unit testing is to check that as individual parts are functioning as expected. Basically Unit testing is typically carried out by the developer.

**Integration testing** – *System Integration Testing* is started after the individual software modules are integrated as a group. A typical software project consists of multiple modules & these are developed by different developers. So in integration testing is focuses to check that after integrating modules Is two modules are communicating with each other or not. It is critical to test every module’s effect on the entire program model. Most of the issues are observed in this type of testing.

**System testing** – This is the first time end to end testing of application on the complete and fully integrated software product before it is launch to the market.

**Acceptance testing** – User acceptance is a type of testing performed by the Client to certify the system with respect to the requirements that was agreed upon.  This is beta testing of the product & evaluated by the actual end users. The main purpose of this testing is to validate the end to end business flow

## What do you verify in System Testing?

System testing involves testing the software code for following

* **Testing the fully integrated applications** including external peripherals in order to check how components interact with one another and with the system as a whole. This is also called End to End scenario testing..
* Verify thorough **testing of every inpu**t in the application to check for desired outputs.
* Testing of the **user's experience** with the application. .

## Types of System Testing

There are more than 50 types of System Testing.. Below we have listed types of system testing a large software development company would typically use

1. **Usability Testing –**

a. What is usability testing?

is a technique used in [user-centered](http://en.wikipedia.org/wiki/User-centered_design) [interaction design](http://en.wikipedia.org/wiki/Interaction_design) to evaluate a product by testing it on users. This can be seen as an irreplaceable [usability](http://en.wikipedia.org/wiki/Usability) practice, since it gives direct input on how real users use the system.[[1]](http://en.wikipedia.org/wiki/Usability_testing#cite_note-1) This is in contrast with [usability inspection](http://en.wikipedia.org/wiki/Usability_inspection) methods where experts use different methods to evaluate a user interface without involving users. Usability testing mainly focuses on the user's-ease to use the application, flexibility in handling controls and ability of the system to meet its objectives.  Examples of products that commonly benefit from usability testing are [foods](http://en.wikipedia.org/wiki/Food), consumer products, [web sites](http://en.wikipedia.org/wiki/Web_design) or web applications, [computer interfaces](http://en.wikipedia.org/wiki/User_interface), documents, and devices.

b. Method:

Setting up a usability test involves carefully creating a [scenario](http://en.wikipedia.org/wiki/Scenario), or realistic situation, wherein the person performs a list of tasks using the product being tested while observers watch and take notes.

1. **Load Testing -** Load testing is necessary to know that a software solution will perform under real life loads.

is the process of putting demand on a system or device and measuring its response. Load testing is performed to determine a system’s behavior under both normal and anticipated peak load conditions. It helps to identify the maximum operating capacity of an application as well as any bottlenecks and determine which element is causing degradation. When the load placed on the system is raised beyond normal usage patterns, in order to test the system's response at unusually high or peak loads, it is known as [stress testing](http://en.wikipedia.org/wiki/Stress_testing). The load is usually so great that error conditions are the expected result, although no clear boundary exists when an activity ceases to be a load test and becomes a stress test.

2.1 Load testing tools:

A[pache JMeter](http://en.wikipedia.org/wiki/Apache_JMeter), [BlazeMeter](http://en.wikipedia.org/wiki/BlazeMeter), [Blitz](http://en.wikipedia.org/wiki/Blitz_(software)), [Gatling](http://en.wikipedia.org/wiki/Gatling_(software)), [LoadRunne](http://en.wikipedia.org/wiki/HP_LoadRunner)r.

**Regression Testing** - Regression testing involves testing done to make sure none of the changes made over the course of the development process have caused new bugs. It also makes sure no old bugs appear from the addition of new software modules over time.  is a type of [software testing](http://en.wikipedia.org/wiki/Software_testing) that seeks to uncover new [software bugs](http://en.wikipedia.org/wiki/Software_bug), or [*regressions*](http://en.wikipedia.org/wiki/Software_regression), in existing [functional](http://en.wikipedia.org/wiki/Functional_testing) and [non-functional](http://en.wikipedia.org/wiki/Non-functional_testing) areas of a system after changes such as enhancements, [patches](http://en.wikipedia.org/wiki/Patch_(computing)) or [configuration](http://en.wikipedia.org/wiki/Configuration_file) changes, have been made to them.

The intent of regression testing is to ensure that changes such as those mentioned above have not introduced new faults.[[1]](http://en.wikipedia.org/wiki/Regression_testing#cite_note-1) One of the main reasons for regression testing is to determine whether a change in one part of the software affects other parts of the software.[[2]](http://en.wikipedia.org/wiki/Regression_testing#cite_note-2)

Method:

Common methods of regression testing include rerunning previously completed tests and checking whether program behavior has changed and whether previously fixed faults have re-emerged. Regression testing can be performed to test a system efficiently by systematically selecting the appropriate minimum set of tests needed to adequately cover a particular change.

1. **Recovery Testing -**  is the activity of testing how well an [application](http://en.wikipedia.org/wiki/Application_software) is able to recover from [crashes](http://en.wikipedia.org/wiki/Crash_(computing)), hardware failures and other similar problems. Recovery testing is done to demonstrate a software solution is reliable, trustworthy and can successfully recoup from possible crashes.

Examples of recovery testing:

1. While an application is running, suddenly restart the computer, and afterwards check the validness of the application's data integrity.
2. While an application is receiving data from a [network](http://en.wikipedia.org/wiki/Computer_network), unplug the connecting cable. After some time, plug the cable back in and analyze the application's ability to continue receiving data from the point at which the network connection disappeared.
3. Restart the system while a [browser](http://en.wikipedia.org/wiki/Web_browser) has a definite number of sessions. Afterwards, check that the [browser](http://en.wikipedia.org/wiki/Web_browser) is able to recover all of them.
4. **Migration Testing -** Migration testing is done to ensure that the software can be moved from older system infrastructures to current system infrastructures without any issues.
5. **Functional Testing -** Also known as functional completeness testing, functional testing involves trying to think of any possible missing functions. Testers might make a list of additional functionalities that a product could have to improve it during functional testing.

is a [quality assurance](http://en.wikipedia.org/wiki/Quality_assurance) (QA) process[[1]](http://en.wikipedia.org/wiki/Functional_testing#cite_note-1) and a type of [black box testing](http://en.wikipedia.org/wiki/Black_box_testing) that bases its test cases on the specifications of the software component under test. Functions are tested by feeding them input and examining the output, and internal program structure is rarely considered (not like in [white-box testing](http://en.wikipedia.org/wiki/White-box_testing)).[[2]](http://en.wikipedia.org/wiki/Functional_testing#cite_note-KanerFalkNguyen1999-2)Functional Testing usually describes *what* the system does.

Functional testing typically involves five steps[[*citation needed*](http://en.wikipedia.org/wiki/Wikipedia:Citation_needed)]

1. The identification of functions that the software is expected to perform
2. The creation of input data based on the function's specifications
3. The determination of output based on the function's specifications
4. The execution of the test case
5. The comparison of actual and expected outputs
6. To check whether the application works as per the customer need.
7. **Hardware/Software Testing -** IBM refers to Hardware/Software testing as "HW/SW Testing". This is when the tester focuses his/her attention on the interactions between the hardware and software during system testing.