Software Testing Methodologies - Learn The Methods & Tools

# **Testing Methodologies**

## What are Testing Methodologies?

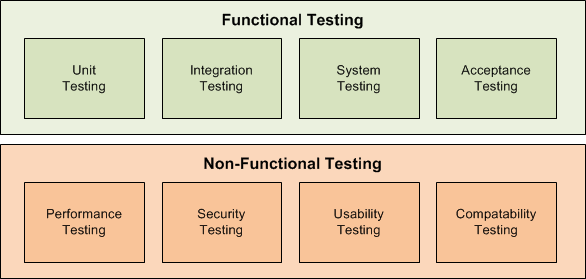
Testing methodologies are the strategies and approaches used to test a particular product to ensure it is fit for purpose. Testing methodologies usually involve testing that the product works in accordance with its specification, has no undesirable side effects when used in ways outside of its design parameters and worst case will fail-safely (e.g. a nuclear reactor will shut down on failure).

## What are Software Testing Methodologies?

Software testing methodologies are the different approaches and ways of ensuring that a software application in particular is fully tested. Software testing methodologies encompass everything from unit testing individual modules, integration testing an entire system to specialized forms of testing such as security and performance.

## Importance of Testing Methodologies

As software applications get ever more complex and intertwined and with the large number of different platforms and devices that need to get tested, it is more important than ever to have a robust [testing methodology](http://www.inflectra.com/SpiraTest/Highlights.aspx?feature=Test-Case-Management) for making sure that software products/systems being developed have been fully tested to make sure they [meet their specified requirements](http://www.inflectra.com/SpiraTest/Highlights.aspx?feature=Requirements-Management) and can successfully operate in all the anticipated environments with the required usability and security.

[](https://www.inflectra.com/GraphicsViewer.aspx?url=~/Ideas/Topics/testing-methodologies.doc&name=wordml://03000001.png)

This page describes the various components of a thorough [testing methodology](http://www.inflectra.com/SpiraTest/Highlights.aspx?feature=Test-Case-Management) and illustrates how [SpiraTest](http://www.inflectra.com/SpiraTest/)is best suited to help you implement and manage them on your projects.

[SpiraTest®](http://www.inflectra.com/SpiraTest) manages your project's requirements, test cases, bugs and issues in one integrated environment, with full traceability throughout the testing lifecycle.

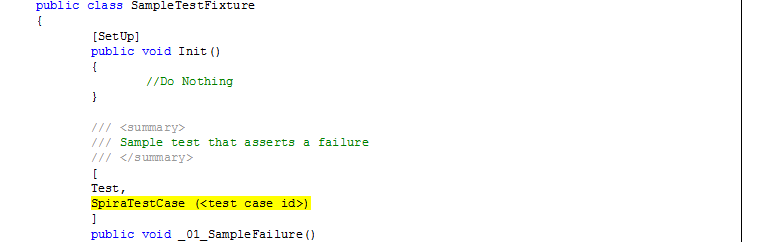
* It is a complete solution that includes [requirements management](http://www.inflectra.com/SpiraTest/Highlights.aspx?feature=Requirements-Management), [test](http://www.inflectra.com/SpiraTest/Highlights.aspx?feature=Test-Case-Management), [release management](http://www.inflectra.com/SpiraTest/Highlights.aspx?feature=Release-Management) and [defect tracking](http://www.inflectra.com/SpiraTest/Highlights.aspx?feature=Defect-Tracking) all fully integrated from day one.
* Highly intuitive web application that provides a complete picture of a project’s status and health yet requires only a web-browser.
* Ability to leverage your existing technology investments. [SpiraTest](http://www.inflectra.com/SpiraTest)integrates with many [automated testing solutions](http://www.inflectra.com/SpiraTest/Integrations.aspx?type=Automated-Testing-Tools) and third-party [defect-management systems](http://www.inflectra.com/SpiraTest/Integrations.aspx?type=Bug-Tracking-Tools).

## Functional Testing

The functional testing part of a testing methodology is typically broken down into four components - unit testing, integration testing, system testing and acceptance testing – usually executed in this order. Each of them is described below:

## Unit Testing

The [Unit testing](http://www.inflectra.com/SpiraTest/Integrations.aspx?type=Unit-Test-Frameworks) part of a testing methodology is the testing of individual software modules or components that make up an application or system. These tests are usually written by the developers of the module and in a test-driven-development methodology (such as Agile, Scrum or XP) they are actually written before the module is created as part of the specification. Each module function is tested by a specific unit test fixture written in the same programming language as the module.

[](https://www.inflectra.com/GraphicsViewer.aspx?url=~/Ideas/Topics/testing-methodologies.doc&name=wordml://03000002.png)

[SpiraTest](http://www.inflectra.com/SpiraTest/)includes support for the management and execution of [automated unit tests](http://www.inflectra.com/SpiraTest/Integrations.aspx?type=Unit-Test-Frameworks). With its library of plugins and extensions for all of the major unit test frameworks (xUnit, JUnit, NUnit, PyUnit, etc.), SpiraTest allows the test manager to ensure that there is full coverage of all program code and that all unit tests have passed.

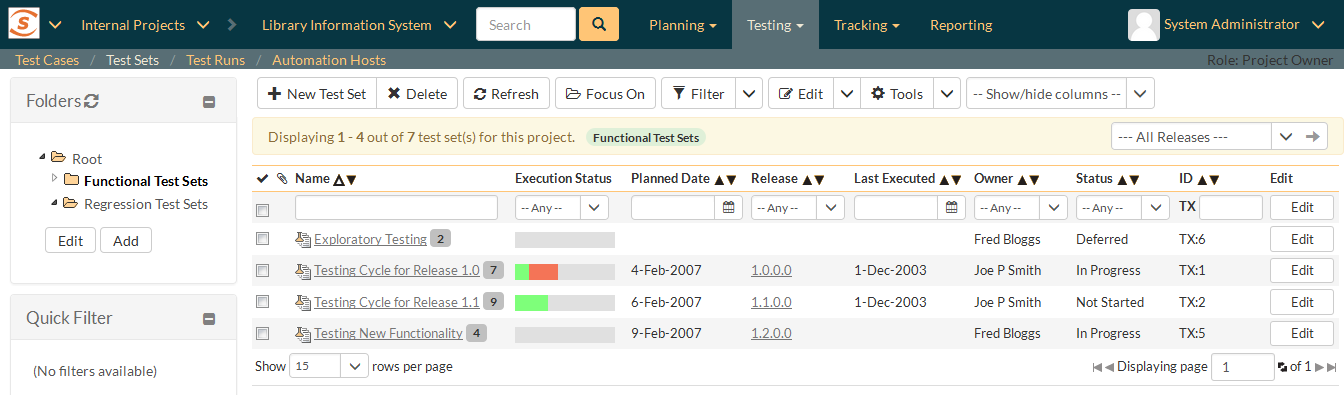
## Integration Testing

The Integration testing part of a testing methodology is the testing of the different modules/components that have been successfully unit tested when integrated together to perform specific tasks and activities (also known as scenario testing). This testing is usually done with a combination of [automated functional tests](http://www.inflectra.com/SpiraTest/Highlights.aspx?feature=Automated-Testing) and manual testing depending on how easy it is to create automated tests for specific integrated components.

[SpiraTest](http://www.inflectra.com/SpiraTest/)includes support for storing, managing and coordinating integration tests across various modules and components. With its library of plugins and extensions for different [automated functional testing](http://www.inflectra.com/SpiraTest/Integrations.aspx?type=Automated-Testing-Tools) tools including our [Rapise](http://www.inflectra.com/Rapise)automation platform, SpiraTest is the perfect solution for managing your integration testing.

## System Testing

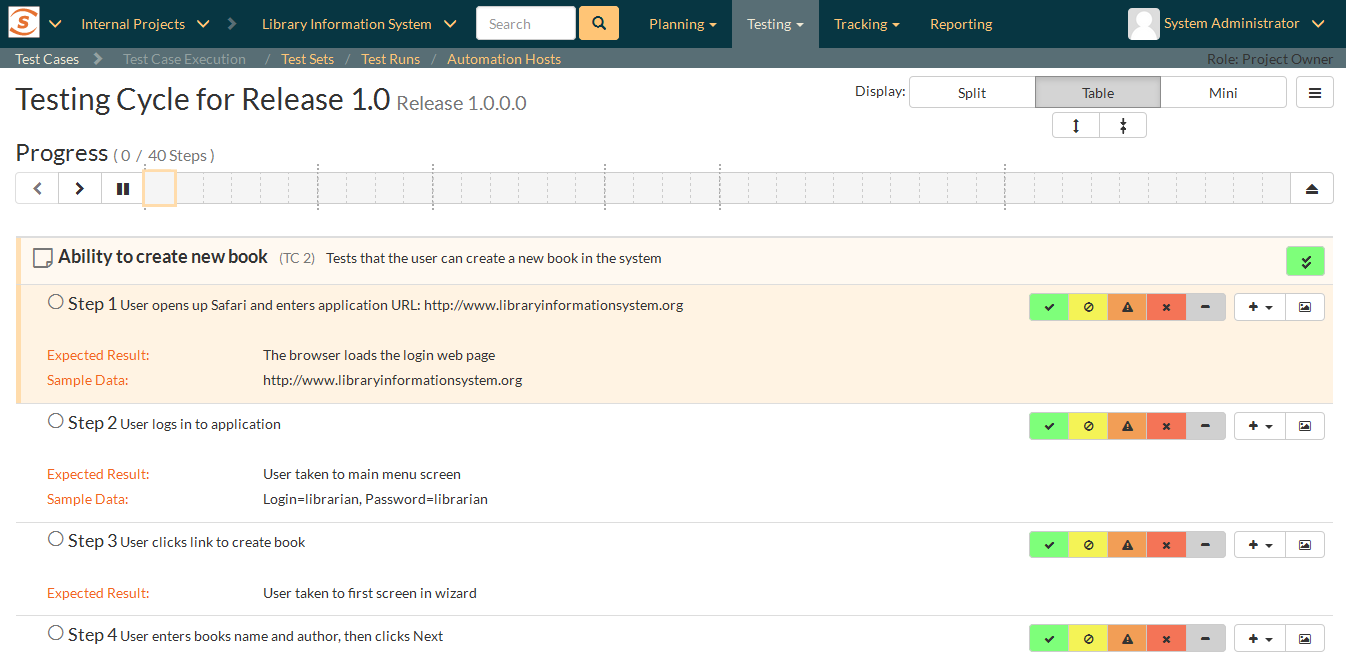
The system testing part of a testing methodology involves testing the entire system for errors and bugs. This test is carried out by interfacing the hardware and software components of the entire system (that have been previously unit tested and integration tested), and then testing it as a whole. This testing is listed under the black-box testing method, where the software is checked for user-expected working conditions as well as potential exception and edge conditions.

[](https://www.inflectra.com/GraphicsViewer.aspx?url=~/Ideas/Topics/testing-methodologies.doc&name=wordml://03000003.png)

[SpiraTest](http://www.inflectra.com/SpiraTest/)includes support for storing, managing and coordinating system tests across all the modules and components that make up a system. SpiraTest support [data-driven testing](http://www.inflectra.com/SpiraTest/Highlights.aspx?feature=Automated-Testing) where test cases are defined with input [parameters](http://www.inflectra.com/SpiraTest/Highlights.aspx?feature=Test-Case-Management)and different combinations of test data can be passed to the manual and automated tests. This ensures that both expected and exception cases can be tested using the same test frameworks.

## Acceptance Testing

The acceptance testing part of a testing methodology is the final phase of functional software testing and involves making sure that all the product/project requirements have been met and that the end-users and customers have tested the system to make sure it operates as expected and meets all their defined requirements:

[](https://www.inflectra.com/GraphicsViewer.aspx?url=~/Ideas/Topics/testing-methodologies.doc&name=wordml://03000004.png)

[SpiraTest](http://www.inflectra.com/SpiraTest/)provides a powerful [manual testing solution](http://www.inflectra.com/SpiraTest/Highlights.aspx?feature=Manual-Testing) that helps you coordinate and manage you acceptance testing activities, with all the test scripts, assignments, test results and associated [defects/bugs](http://www.inflectra.com/SpiraTest/Highlights.aspx?feature=Defect-Tracking) all tracked in a single unified system. With SpiraTest’s [requirements test coverage](http://www.inflectra.com/SpiraTest/Highlights.aspx?feature=Requirements-Management) capabilities you can validate that all of your requirements have been fully tested to your customer’s satisfaction.

## Non-Functional Testing

In most [testing methodologies](http://www.inflectra.com/SpiraTest/Default.aspx), functional testing involves testing the application against the business requirements. Functional testing is done using the functional specifications provided by the client or by using the design specifications like use cases provided by the design team.

On the other hand, non-functional testing involves testing the application against the non-functional requirements, which typically involve measuring/testing the application against defined technical qualities (also known as the ‘-ilities’ because they all end in ‘-ility), for example: vulnerability, scalability, usability. Some examples of non-functional testing are described below:

## Performance, Load, Stress Testing

There are several different types of performance testing in most testing methodologies, for example: performance testing is measuring how a system behaves under an increasing load (both numbers of users and data volumes), load testing is verifying that the system can operate at the required response times when subjected to its expected load, and stress testing is finding the failure point(s) in the system when the tested load exceeds that which it can support.

[](https://www.inflectra.com/GraphicsViewer.aspx?url=~/Ideas/Topics/testing-methodologies.doc&name=wordml://03000005.png)

[SpiraTest](http://www.inflectra.com/SpiraTest/)includes support for storing, managing and coordinating your performance, load and stress testing activities. With its library of plugins and extensions for different [automated performance testing](http://www.inflectra.com/SpiraTest/Integrations.aspx?type=Automated-Testing-Tools) tools including LoadRunner, JMeter and NeoLoad, SpiraTest is the perfect solution for managing your performance testing.

## Security, Vulnerability Testing

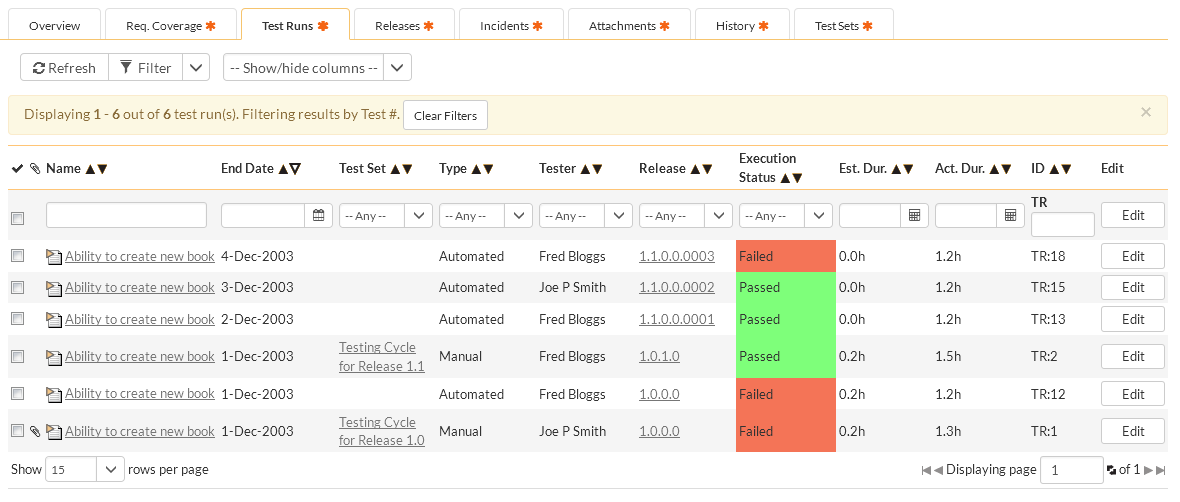
Previously, security was something that was tested after-the-fact. With the rise in cyber-crime and the awareness of the risks associated with software vulnerabilities, application security is now something that needs to be designed and developed at the same time as the desired business functionality. Security testing tests the software for confidentiality, integrity, authentication, availability, and non-repudiation. Individual tests are conducted to prevent any unauthorized access to the software code.

## Usability Testing

The usability testing part of a testing methodology looks at the end-user usability aspect of the software. The ease with which a user can access the product forms the main testing point. Usability testing looks at five aspects of testing, - learnability, efficiency, satisfaction, memorability, and errors.

## Compatibility Testing

The compatibility part of a testing methodology tests that the product or application is compatible with all the specified operating systems, hardware platforms, web browsers, mobile devices, and other designed third-party programs (e.g. browser plugins). Compatibility tests check that the product works as expected across all the different hardware/software combinations and that all functionality is consistently supported.

[](https://www.inflectra.com/GraphicsViewer.aspx?url=~/Ideas/Topics/testing-methodologies.doc&name=wordml://03000006.png)

[SpiraTest](http://www.inflectra.com/SpiraTest/)makes the managing and tracking of your [cross-platform testing](http://www.inflectra.com/SpiraTest/Highlights.aspx?feature=Test-Case-Management) easy, it enabled you to quickly configure different test plans for the different hardware/software combinations and make sure that all of the requirements and test cases have been executed (and passed) against all of the specified combinations and configurations.

## Functional vs. Non-functional Testing

The goal of utilizing numerous testing methodologies in your development process is to make sure your software can successfully operate in multiple environments and across different platforms. These can typically be broken down between functional and non-functional testing. Functional testing involves testing the application against the business requirements. It incorporates all test types designed to guarantee each part of a piece of software behaves as expected by using uses cases provided by the design team or business analyst. These testing methods are usually conducted in order and include:

* Unit testing
* Integration testing
* System testing
* Acceptance testing

Non-functional testing methods incorporate all test types focused on the operational aspects of a piece of software. These include:

* Performance testing
* Security testing
* Usability testing
* Compatibility testing

The key to releasing high quality software that can be easily adopted by your end users is to build a robust [testing framework](https://smartbear.com/learn/automated-testing/test-automation-frameworks/) that implements both functional and non-functional software testing methodologies.

## Unit Testing

Unit testing is the first level of testing and is often performed by the developers themselves. It is the process of ensuring individual components of a piece of software at the code level are functional and work as they were designed to. Developers in a test-driven environment will typically write and run the tests prior to the software or feature being passed over to the test team. Unit testing can be conducted manually, but automating the process will speed up delivery cycles and expand test coverage. Unit testing will also make debugging easier because finding issues earlier means they take less time to fix than if they were discovered later in the testing process. TestLeft is a tool that allows advanced testers and developers to shift left with the fastest test automation tool embedded in any IDE.

### Start Shifting Left and Automate now with ****TestLeft****

[Start Free Trial](https://smartbear.com/product/testleft/free-trial)

## Integration Testing

After each unit is thoroughly tested, it is integrated with other units to create modules or components that are designed to perform specific tasks or activities. These are then tested as group through integration testing to ensure whole segments of an application behave as expected (i.e, the interactions between units are seamless). These tests are often framed by user scenarios, such as logging into an application or opening files. Integrated tests can be conducted by either developers or independent testers and are usually comprised of a combination of automated functional and manual tests.

## System Testing

System testing is a black box testing method used to evaluate the completed and integrated system, as a whole, to ensure it meets specified requirements. The functionality of the software is tested from end-to-end and is typically conducted by a separate testing team than the development team before the product is pushed into production.

## Acceptance Testing

Acceptance testing is the last phase of functional testing and is used to assess whether or not the final piece of software is ready for delivery. It involves ensuring that the product is in compliance with all of the original business criteria and that it meets the end user’s needs. This requires the product be tested both internally and externally, meaning you’ll need to get it into the hands of your end users for beta testing along with those of your QA team. Beta testing is key to getting real feedback from potential customers and can address any final usability concerns.

## Performance Testing

[Performance testing](https://smartbear.com/learn/performance-testing/what-is-load-testing/) is a non-functional testing technique used to determine how an application will behave under various conditions. The goal is to test its responsiveness and stability in real user situations. Performance testing can be broken down into four types:

* **Load testing** is the process of putting increasing amounts of simulated demand on your software, application, or website to verify whether or not it can handle what it’s designed to handle.
* **Stress testing** takes this a step further and is used to gauge how your software will respond at or beyond its peak load. The goal of stress testing is to overload the application on purpose until it breaks by applying both realistic and unrealistic load scenarios. With stress testing, you’ll be able to find the failure point of your piece of software.
* **Endurance testing,** also known as soak testing, is used to analyze the behavior of an application under a specific amount of simulated load over longer amounts of time. The goal is to understand how your system will behave under sustained use, making it a longer process than load or stress testing (which are designed to end after a few hours). A critical piece of endurance testing is that it helps uncover memory leaks.
* **Spike testing** is a type of load test used to determine how your software will respond to substantially larger bursts of concurrent user or system activity over varying amounts of time. Ideally, this will help you understand what will happen when the load is suddenly and drastically increased.

## Security Testing

With the rise of cloud-based testing platforms and cyber attacks, there is a growing concern and need for the security of data being used and stored in software. Security testing is a non-functional software testing technique used to determine if the information and data in a system is protected. The goal is to purposefully find loopholes and security risks in the system that could result in unauthorized access to or the loss of information by probing the application for weaknesses. There are multiple types of this testing method, each of which aimed at verifying six basic principles of security:

1. Integrity
2. Confidentiality
3. Authentication
4. Authorization
5. Availability
6. Non-repudiation

## Usability Testing

Usability testing is a testing method that measures an application’s ease-of-use from the end-user perspective and is often performed during the system or acceptance testing stages. The goal is to determine whether or not the visible design and aesthetics of an application meet the intended workflow for various processes, such as logging into an application. Usability testing is a great way for teams to review separate functions, or the system as a whole, is intuitive to use.

## Compatibility Testing

Compatibility testing is used to gauge how an application or piece of software will work in different environments. It is used to check that your product is compatible with multiple operating systems, platforms, browsers, or resolution configurations. The goal is to ensure that your software’s functionality is consistently supported across any environment you expect your end users to be using.

## Testing With TestComplete

[TestComplete](https://smartbear.com/product/testcomplete/overview/) is our robust automated GUI testing tool that excels in compatibility and integration testing. It helps QA teams create and run tests across desktop, mobile, and web applications – enabling testing professionals to speed up delivery cycles and improve software quality. Testcomplete comes with built-in support for various test environments, integrations to performance testing tools, as well as support for developer friendly SCMs, allowing you to seamlessness integrate it into your development process. Using TestComplete will enable you to build a robust testing framework that utilizes the broad spectrum of available software testing methodologies.

**What are the Software Development and Testing Methodologies?**

Testing is an essential part of the Software Development Process. A robust and stable software product can be delivered with the use of standard testing methodologies that will help to predict the timeline of the software system.

A software application may turn even more complex with a large number of platforms and devices. More importantly, it is required to ensure whether they meet the specified requirements and can be efficiently installed and operated on the user’s machine or not.

With the means of [security](https://www.softwaretestinghelp.com/how-to-test-application-security-web-and-desktop-application-security-testing-techniques/), [compatibility](https://www.softwaretestinghelp.com/software-compatibility-testing/), and usability, a software product should be tested by using the proper testing methodology.

**In this article*,* we will discuss on what is meant by Testing Methodologies, how it differs from testing strategies, and types of Software Testing Methods in detail.**

**What You Will Learn:**[[show](https://www.softwaretestinghelp.com/software-development-testing-methodologies/)]

### Meaning of Testing Methodologies

Methodologies can be considered as the set of testing mechanisms used in software development lifecycle from Unit Testing to System Testing. Selecting an appropriate testing methodology is considered to be the core of the testing process.

### Testing Techniques

Basically, there are 3 testing methodologies which are used for testing. They are White Box Testing, Black Box Testing, and Grey Box Testing. These are also called as ***Testing Techniques***. Each of the testing technique is briefed below for your better understanding.

#### **#1) White Box Testing:**

[White box testing technique](https://www.softwaretestinghelp.com/white-box-testing-techniques-with-example/) is used to examine the program structure and business logic, it validates the code or program of an application. It is also called as Clear Box Testing, Glass Box Testing or Open Box Testing.

**White Box Testing Techniques include:**

* **Statement Coverage:** Examines all the programming statements.
* **Branch Coverage:** Series of running tests to ensure if all the branches are tested.
* **Path Coverage:**Tests all the possible paths to cover each statement and branch.

#### **#2) Black Box Testing:**

[Black Box testing method](https://www.softwaretestinghelp.com/black-box-testing/) is used to test the functionality of an application based on the requirement specification. Unlike White Box Testing it does not focus on internal structure/code of the application.

**Black Box Techniques include:**

* [Boundary Value analysis](https://www.softwaretestinghelp.com/what-is-boundary-value-analysis-and-equivalence-partitioning/)
* Equivalence Partitioning(Equivalence Class Partitioning)
* Decision Tables
* Domain Tests
* State Models
* Exploratory Testing(Requires less preparation and also helps to find the defects quickly).

#### **#3) Grey Box Testing:**

This method of testing is performed with less information about the internal structure of an application. Generally, this is performed like Black Box Testing only but for some critical areas of application, White Box Testing is used.

### Models in SDLC

Selecting proper testing methodologies also incorporate with choosing a proper model in SDLC.

**The models include:**

* Waterfall model
* V model
* Agile model
* Spiral model
* RAD

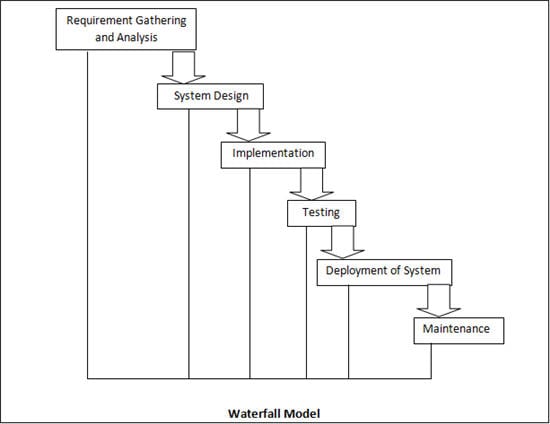
**Let’s have closer look at each Software Development Methodologies with a brief explanation.**

#### **#1) Waterfall Model**

[Waterfall model](https://www.softwaretestinghelp.com/what-is-sdlc-waterfall-model/) is the basic life cycle model which was developed by Winston Royce in 1970. This model represents multiple stages or processes in a sequential manner that flows progressively downward.

This approach is useful when requirements are well known, technology is understood and the resources with required expertise are available.

**Waterfall model is defined by the following stages:**

[](https://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2017/11/Waterfall-Model.jpg)

* **Requirement Gathering and Analysis:**Capture and analyze all the requirements and make sure whether they are testable or not.
* **System Design:**Create and document design based on requirement analysis. Define the hardware and software requirements.
* **Implementation:**Create robust code for components as per the design and integrate them.
* **System Testing:**Integrated components form a whole system, this phase is performed to ensure whether the system is working as per the requirements, tracking and reporting the testing progress.
* **System Deployment:**Make sure if system is stable with zero bugs, all test criteria had been  
  met, ensure Environment Setup etc.
* **System Maintenance:**Makes sure if the application is working efficiently as per the requirement with the suitable environment. In case a defect is found then that should be fixed and deployed (updated) in the environment.

**Advantages of Waterfall Model:**

* Simple and easy to understand.
* Easy to manage as each phase has its own specific deliverables.
* Overlapping of stages is avoided.
* Good for small projects.

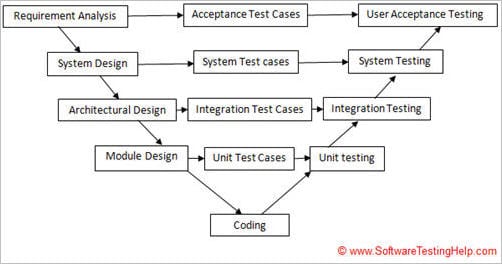
**Disadvantages of Waterfall Model:**

* Increase in the amount of risk and uncertainty.
* Once entered into Testing phase, cannot change anything in the previous stages **e.g** Design and Coding etc.
* Not good for complex and large projects.
* Not suitable where the requirements keep changing.

#### **#2) V Model**

[V Model](https://www.softwaretestinghelp.com/what-is-stlc-v-model/) is an extension of Waterfall Model where the process execution takes place in a sequential style in V-Shape and is also known as Verification and Validation Model. In this approach, there exists a directly associated testing phase in every single phase of the development cycle.

It has been proven beneficial and cost-efficient than the waterfall model as the testing is performed at each development phase rather than at the end of the development cycle.

[](https://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2017/12/v-model.jpg)

**V Model is classified into 3 Phases.**

* Verification Phase
* Coding Phase
* Validation Phase

**a) Verification Phase**:

* **Business Requirement Analysis:** Communicate with the customer to understand their expectations and requirements.
* **System Design:** Design complete system and its components along with the hardware and software requirements.
* **Architectural Design:**In this phase architectural specifications are captured. This also known as high-level Design.
* **Module Design:** This is also known as Low-Level Design, Detailed internal design for all the specified system modules.

**b) Coding Phase:**

This phase contains actual coding phase in the development lifecycle. Programming languages should be chosen based on the system and architectural design specified in the previous phase technology platform. Coding is performed according to the standards and guidelines that are pre-defined.

**c) Validation Phase**:

* **Unit Testing:** Performed on an individual module to eliminate the bugs at the early stage.
* **Integration Testing:** Performed to test the communication between different modules in the system.
* **System Testing:**[System Testing](https://www.softwaretestinghelp.com/system-testing/) is performed on a system as a whole.
* **Acceptance Testing:** This is associated with the business requirements. It is performed in a user environment from the user’s point of view.

**Advantages of V model**

* Simple, easy to use and understand.
* Overlapping is avoided as phases are executed one at a time.
* Easy to manage and suitable for small projects.

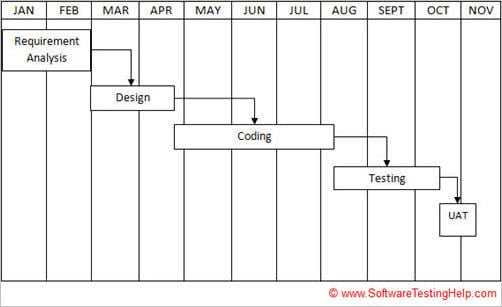
**Disadvantages of V Model are more or less similar to the disadvantages of Waterfall model.**

#### **#3) Agile Model**

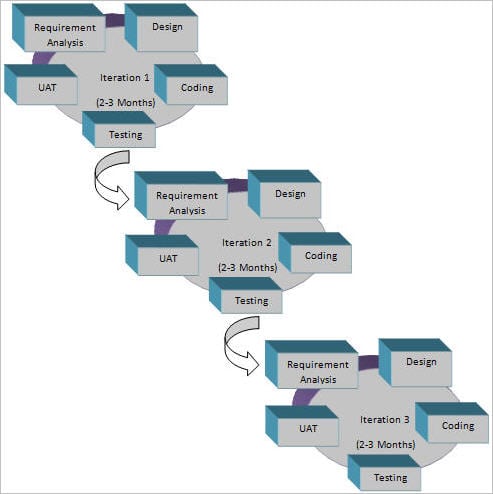
[Agile Model](https://www.softwaretestinghelp.com/agile-scrum-methodology-for-development-and-testing/) shows an iterative and incremental approach. This approach breaks the product into small incremental units to provide iterations. Then each iteration involves steps like Planning, Requirement Analysis, Design, Coding, Unit Testing, Acceptance Testing etc.

This approach also allows continuous interaction with the customer for their feedback and corrections in the requirements at regular intervals.

**Following diagram will help you to understand Agile Model approach more precisely:**

[](https://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2017/12/agile-model.jpg)

**Following image will show the iteration cycle in Agile Model:**

[](https://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2017/12/iteration-cycle-in-agile-model.jpg)

**Advantages of Agile model:**

* A realistic approach to software development.
* Promotes teamwork.
* Eliminates mismatch between requirements and test cases.
* Rapid and requires minimum amount of resources.
* Suitable for large and long-term projects.
* Good for changing requirements.
* Easy to manage.

**Disadvantages of Agile model:**

* Not suitable for complex projects.
* Requires heavy amount of interaction with the customer which may cause delay.
* Misguidance of requirements may cause the incorrect development of the software product.
* Increased maintainability risk.
* Handover to another team may be quite challenging.

#### **#4) Spiral Model**

The spiral model incorporates iterative development approach along with the systematic approach of the waterfall model. It is similar to the incremental model and emphasis on Risk Analysis.

**Spiral Model has four stages:**

* Planning Phase
* Risk Analysis
* Engineering Phase
* Evaluation Phase

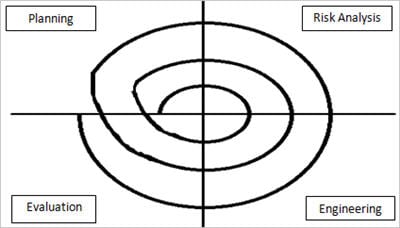
**1) Planning Phase:**In this phase, the requirements are gathered and reviewed to finalize the test case.

**2) Risk Analysis:**This stage includes identifying, monitoring and estimating management risks. Requirements are analyzed to identify the risks using techniques like brainstorming, walkthrough etc.

**3) Engineering Phase:**In this phase, the software is developed and tested at the end.

**4) Evaluation Phase:**This is the last stage where a customer evaluates the output of a project and gives their feedback for either next spiral or approval.

**Pictorial representation of Spiral Model:**

[](https://cdn.softwaretestinghelp.com/wp-content/qa/uploads/2017/11/Spiral-Model.jpg)

**When to use Spiral model:**

* For high-risk projects.
* When the requirements are complex.
* If a project is large.
* Have sufficient amount of time for getting user’s feedback for the next spiral.
* Requires significant changes due to research and exploration.
* Users are not sure of their needs.

**Advantages of Spiral Model:**

* Avoidance of risk as it involves a high amount of risk analysis.
* Rapid Development.
* Changes in requirements are accommodated easily.
* Requirements can be acquired more accurately.

**Disadvantages of Spiral model:**

* Complex management.
* Not suitable for small projects.
* May involve no. of spirals(indefinite).
* Costly.
* Requires high amount of risk analysis and expertise for their project's success.

#### **#5) RAD Model**

Rapid Application Development(RAD) is a type of incremental model. In this approach, components are developed in parallel.

This is a rapid approach and it can give a fast product to the customer to provide feedback.

**Phases in RAD are as follows:**

* **Business Modeling:**Identifies vital information and its flow between various business channels.
* **Data Modeling:**Information gathered in the previous stage is used to define data objects required for the business.
* **Process modeling:**Data objects are converted to get business objective and flow of information.
* **Application Generation:**In this phase, automation tools are used to convert the process model into actual code.
* **Testing and Turnover:**Tests all the components of a system, hence overall testing time is reduced.

**Advantages of RAD Model:**

* Progress can be measured.
* Reduces development time.
* Increased reusability.
* Quick initial reviews.
* Enhances customer feedback.

**Disadvantages of RAD Model:**

* Requires high skilled resources.
* High-cost estimation.
* Not applicable for cheaper projects.
* High dependency on modeling skills.
* Only a modularized system can be built using RAD.

### Difference between Testing Methodologies & Testing Strategies

The answer to this is not much complex as there is a simple difference between both.

**Testing Methodologies** are the methods or approaches to testing that includes from Unit testing through System Testing.

**Testing Strategies**is an overview of the key issues that occur in the testing process and is to be taken into consideration by the project manager, a team of developers and testers.

The above-discussed Software Testing Methods are used to implement n number of testing strategies.

**Some of them are listed below:**

**1) Unit Testing:**

* Focuses on very small functional units.
* The simplest way to check smallest units for isolation.
* Generally performed by developers.

**2) Integration Testing:**

* This is the next step to be performed on the developer's side.
* Provide mechanism to test interaction, inter-operation, and communication between the different modules of software

**3) Functional Testing:**

It is used to check the functionalities of a software system i.e. output to the given input.

**4) Regression Testing:**

Checks if the bug fixing has happened at one place so that the complex functionalities should not cause any change in another core area.

**5) System Testing:**

* Testing all the integrated modules as a collective system.
* Combines multiple features into end-to-end scenarios.

**6) Performance Testing:**

Tests the performance of the application in critical situations like transferring big sized file, concurrent users access to the system, configuration failure etc.

**7) Acceptance Testing**:

* Generally Final level of testing where software product is examined as users perspective by testers
* The result of this step is subjective and takes a little to find exact issue

### Conclusion:

Choosing a proper Testing Methodology is the action or set of actions that lies at the core of the testing process. This may even be a versatile activity that changes according to the business requirements and timeline of the software product.

However, one can choose single or even multiple Software Development and Testing Methodologies to have a more flexible and efficient end product that satisfies the customer’s needs and expectation in the desired or less time limit.

**What is Software Testing Methodology?**

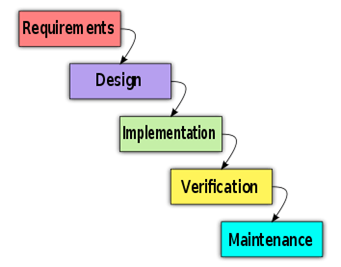
Software Testing Methodology is defined as strategies and testing types used to certify that the Application Under Test meets client expectations. Test Methodologies include functional and non-functional testing to validate the AUT. Examples of Testing Methodologies are [Unit Testing](https://www.guru99.com/unit-testing-guide.html), [Integration Testing](https://www.guru99.com/integration-testing.html), [System Testing](https://www.guru99.com/system-testing.html), [Performance Testing](https://www.guru99.com/performance-testing.html) etc. Each testing methodology has a defined test objective, test strategy, and deliverables.

**Note**: Since Software Testing is an integral part of any Development Methodology, many companies use the term Development Methodologies & Testing Methodologies colloquially. Hence Testing Methodologies could also refer to Waterfall, Agile and other QA models as against the above definition of Testing Methodologies. Discussion on various testing types does not add value to the readers. Hence, we will discuss the different development models.

In this tutorial, you will learn-

* [Waterfall model](https://www.guru99.com/testing-methodology.html#2)
* [Iterative development](https://www.guru99.com/testing-methodology.html#3)
* [Agile methodology](https://www.guru99.com/testing-methodology.html#4)
* [Extreme programming](https://www.guru99.com/testing-methodology.html#5)
* [Which Software Methodology to choose?](https://www.guru99.com/testing-methodology.html#6)
* [How to setup software testing methodologies?](https://www.guru99.com/testing-methodology.html#7)

**Waterfall Model**

[](https://www.guru99.com/images/stories/waterfall.png)

**What is it?**

In the [waterfall model](https://www.guru99.com/what-is-sdlc-or-waterfall-model.html), software development progress through various phases like Requirements Analysis, Design etc -  **sequentially**.

In this model, the next phase begins only when the earlier phase is completed.

**What Is The Testing Approach?**

The first phase in the waterfall model is the requirements phase in which all the project requirements are completely defined before starting the testing. During this phase, the test team brainstorms the scope of testing, test strategy and drafts a detailed test plan.

Only once the design of software is complete, the team will move on to execution of the test cases to ensure that the developed software behaves as it expected.

In this methodology, the testing team proceeds to the next phase only when the previous phase is completed.

**Advantages**

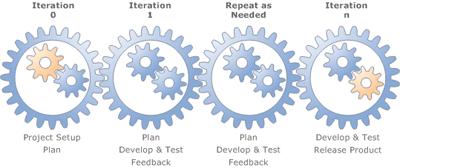
This software Engineering model is very simple to plan and manage. Hence, projects, where requirements are clearly defined and stated beforehand, can be easily tested using a waterfall model.

**Disadvantages**

In the waterfall model, you can begin with the next phase only once the previous phase is completed. Hence, this model cannot accommodate unplanned events and uncertainty.

This methodology is not suitable for projects where the requirements change frequently.

**Iterative development**

[](https://www.guru99.com/images/stories/iterative-model.jpg)

**What is it?**

In this model, a big project is divided into small parts, and each part is subjected to multiple iterations of the waterfall model. At the end of an iteration, a new module is developed or an existing module is enhanced. This module is integrated into the software architecture and the entire system is tested all together

**What is the testing Approach?**

As soon as iteration is completed, the entire system is subjected to testing. Feedback from testing is immediately available and is incorporated in the next cycle. The testing time required in successive iteration can be reduced based on the experience gained from past iterations.

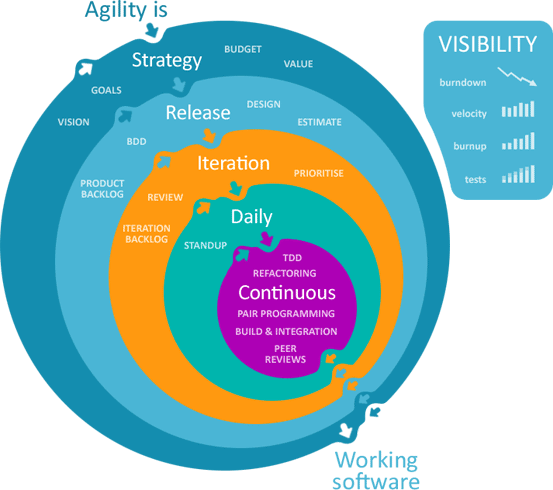
**Advantages**

The main advantage of iterative development is the test feedback is immediately available at the end of each cycle.

**Disadvantages**

This model increases communication overheads significantly since, at the end of each cycle, feedback about deliverables, effort etc must be given.

**Agile methodology**

[](https://www.guru99.com/images/stories/agile_development_model.gif)

**What is it?**

Traditional software development methodologies work on the premise that software requirements remain constant throughout the project. But with an increase in complexity, the requirements undergo numerous changes and continuously evolve. At times, the customer himself is not sure what he wants. Though the iterative model addresses this issue, it's still based on the waterfall model.

In Agile methodology, software is developed in incremental, rapid cycles. Interactions amongst customers, developers and client are emphasized rather than processes and tools. The agile methodology focuses on responding to change rather than extensive planning.

**What Is The Testing Approach?**

 Incremental testing is used in agile development methods and hence, every release of the project is tested thoroughly. This ensures that any bugs in the system are fixed before the next release.

**Advantages**

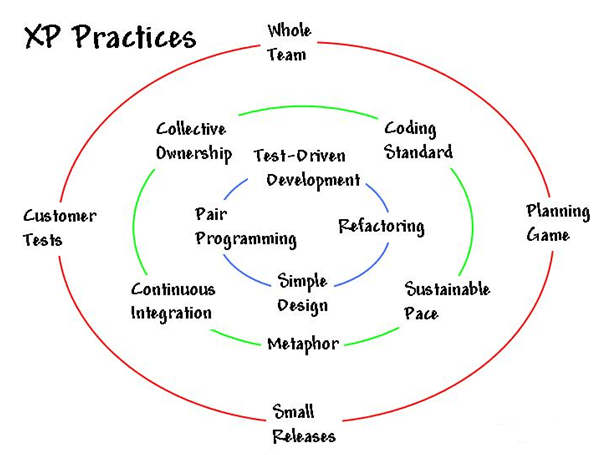
It is possible to make changes in the project at any time to comply with the requirements.

This incremental testing minimizes risks.

**Disadvantages**

Constant client interaction means added time pressure on all stakeholders including the client themselves, software development and test teams.

**Extreme programming**

**[](https://www.guru99.com/images/stories/xtreme%20programming.png)**

**What is it?**

Extreme programming is a type of agile methodology which beliefs in short development cycles. A project is divided into simple engineering tasks. Programmers code a simple piece of software and get back to the customer for feedback. Review points from the customer are incorporated and the developers proceed with the next task.

In extreme programming developers usually, work in pairs.

[Extreme Programming](https://www.guru99.com/agile-scrum-extreme-testing.html) is used in places where customer requirements are constantly changing.

**What Is The Testing Approach?**

Extreme programming follows a Test-driven development which is described as follows -

1. Add a[Test Case](https://www.guru99.com/test-case.html)to the test suite  to verify the new functionality which is yet to be developed
2. Run all the tests and obviously the new test case added must fail since the functionality is not coded yet
3. Write some code to implement the feature/functionality
4. Run the test suite again. This time, the new test case should pass since the functionally has been coded

**Advantages**

Customers having a  vague software design in mind could use extreme programming

Continuous testing and continuous integration of small releases ensure software code is delivered is of high quality

**Disadvantages**

Meetings amongst the software development team and clients add to time requirements.

**Which Software Methodology to choose?**

There are tons of methodologies available for software development and its corresponding testing. Each testing technique and methodology is designed for a specific purpose and has its relative merits and demerits.

Selection of a particular methodology depends on many factors such as the nature of a project, client requirement, project schedule, etc.

From a testing perspective, some methodologies push for testing input early in the development life cycle, while others wait until a working model of the system is ready.

**How to setup software testing methodologies?**

Software testing methodologies should not be set up just for the sake of testing software code. The big picture should be considered and the prime goal of the project should be satisfied with the testing methodology.

**Scheduling**

Realistic scheduling is the key to the implementation of successful testing methodology and the schedule should meet the needs of every member of the team.

**Defined deliverables**

In order to keep all the members of the team on the same page, well-defined deliverables should be provided. The deliverables should contain direct content without any ambiguity.

**Test approach**

Once scheduling is complete and defined deliverables are made available, the testing team should be able to formulate the right test approach. Definition documents and developer meetings should indicate the team about the best test approach that can be used for the project.

**Reporting**

Transparent reporting is very difficult to achieve, but this step determines the effectiveness of the testing approach used in the project.

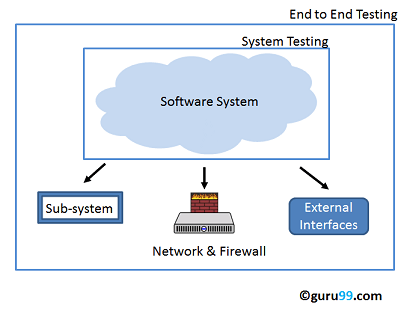
**What is End to End Testing?**

End-to-End Testing is defined as a type Software Testing that not only validates the software system under test but also checks its integration with external interfaces. Hence, the name "**End-to-End**". The purpose of End-to-End Testing is to exercise a complete production-like scenario.

Along with the software system, it also validates batch/data processing from other upstream/downstream systems.

End to End Testing is usually executed after functional and [System Testing](https://www.guru99.com/system-testing.html).

It uses actual production like data and test environment to simulate real-time settings. End-to-End testing is also called Chain Testing

[](https://www.guru99.com/images/End-End-Testing(1).png)

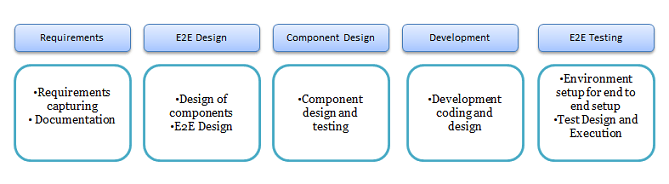
**Why End to End Testing?**

Modern software systems are complex and are interconnected with multiple sub-systems

A sub-system may be different from the current system or may be owned by another organization.  **If anyone of the sub-systems fails, the whole software system could collapse**. This is a major risk and can be avoided by End-to-End testing. End-to-End testing verifies the complete system flow. It increases[Test coverage](https://www.guru99.com/test-coverage-in-software-testing.html)of various sub-systems. It helps detect issues with sub-systems and increases confidence in the overall software product.

**End to End Testing Process:**

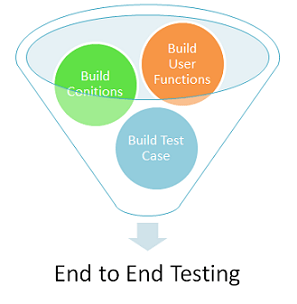
The following diagram gives an overview of the End to End testing process.

[](https://www.guru99.com/images/End-to_end-testing-process(1).png)

The chief activities involved in End to End Testing are -

* Study of an end to end testing requirements
* Test Environment setup and hardware/software requirements
* Describe all the systems and its subsystems processes.
* Description of roles and responsibilities for all the systems
* Testing methodology and standards
* End to end requirements tracking and designing of test cases
* Input and  output data for each system

**How to create End-to-End Test Cases?**

[](https://www.guru99.com/images/End-End-Testing-Test-Cases(1).png)

End to End Testing Design framework consists of three parts

1. Build user functions
2. Build Conditions
3. Build Test Cases

Let's look at them in detail: -

**Build User Functions**

Following activities should be done as a part of build user functions:

* List down the features of the system and their interconnected components
* List the input data, action and the output data for each feature or function
* Identify the relationships between the functions
* Determine whether the function can be reusable or independent

For example -Consider a scenario where you login into your bank account and transfer some money to another account from some other bank (3rdparty sub-system)

1. Login into the banking system
2. Check for the balance amount in the account
3. Transfer some  amount from your account to some other bank account (3rdparty sub-system)
4. Check your latest account balance
5. Logout of the application

**Build Conditions based on User Function**

Following activities are performed as a part of build conditions:

* Building a set of conditions for each user function defined
* Conditions include sequence, timing and data conditions

For example -Checking of more conditions like

**Login Page**

* Invalid User Name and Password
* Checking with valid username and password
* Password strength checking
* Checking of error messages

**Balance Amount**

* Check the current balance after 24 hours. (If the transfer is sent to a different bank)
* Check for the error message if the transfer amount is greater than the current balance amount

**Build a Test Scenario**

Building the[Test Scenario](https://www.guru99.com/test-scenario.html)for the user function defined

In this case,

* Login into the system
* Check of bank balance amount
* Transfer the bank balance amount

**Build Multiple Test cases**

Build one or more test cases for each scenario defined. Test cases may include each condition as a single test case.

**Metrics for End to End testing:**

Following are few of many metrics used for End to End Testing.

* **Test Case preparation status:** It gives Test Case preparation progress against planned
* **Weekly Test Progress-**Provides week-wise details of percentage test completion- Failed, not executed & executed against planned for execution tests.
* **Defects Status & Details-**It gives Percentage of open & closed defects by the week. Also, week-wise defects distribution based on severity and priority
* **Environment Availability -**Total number of hours "up" / Total number of hours scheduled per day for testing

|  |  |
| --- | --- |
| **End to End Testing** | **System Testing** |
| Validates the software system as well as interconnected sub-systems | Validates just the software system as per the requirements specifications. |
| It checks the complete end-to-end process flow. | It checks system functionalities and features. |
| All interfaces, backend systems will be considered for testing | Functional and Non-Functional Testing will be considered for testing |
| It's executed once System Testing is completed. | It's executed after [Integration Testing](https://www.guru99.com/integration-testing.html). |
| End to End testing involves checking external interfaces which can be complex to automate. Hence [Manual Testing](https://www.guru99.com/manual-testing.html) is preferred. | Both Manual and Automation can be performed for System Testing |

**End to End Testing Vs System Testing**

**Conclusion**

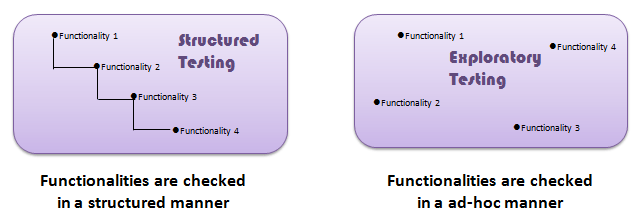
In Software Engineering, End to End Testing is the process verifying a software system along with its sub-systems. The biggest challenge in this testing is to have enough knowledge of the whole system as well as an interconnected sub-system.

**What is Exploratory Testing?**

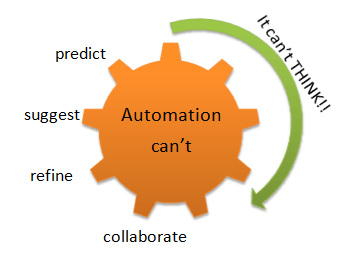
Exploratory testing is all about discovery, investigation, and learning. It emphasizes personal freedom and responsibility of the individual tester. It is defined as a type of testing where Test cases are not created in advance but testers check system on the fly. They may note down ideas about what to test before test execution. The focus of exploratory testing is more on testing as a "thinking" activity.

Under scripted testing, you design test cases first and later proceed with test execution. On the contrary, exploratory testing is a simultaneous process of test design and test execution all done at the same time.

Scripted Test Execution is usually a non-thinking activity where testers execute the test steps and compare the actual results with expected results. Such test execution activity can be automated does not require many cognitive skills.

[](https://www.guru99.com/images/E1.png)

**Though the current trend in software testing is to push for** **automation**, exploratory testing is a new way of thinking. **Automation has its limits**

[](https://www.guru99.com/images/E2.png)

**Differences between Scripted and Exploratory Testing**

|  |  |
| --- | --- |
| **Scripted Testing** | **Exploratory Testing** |
| Directed from requirements | Directed from requirements and exploring during testing |
| Determination of test cases well in advance | Determination of test cases during testing |
| Confirmation of testing with the requirements | Investigation of system or application |
| Emphasizes prediction and decision making | Emphasizes adaptability and learning |
| Involves confirmed testing | Involves Investigation |
| Is about Controlling tests | Is about Improvement of test design |
| Like making a speech - you read from a draft | Like making a conversation - it's spontaneous |
| The script is in control | The tester's mind is in control |

Exploratory testing -

* Is not random testing but it is ad-hoc testing with a purpose of find bugs
* Is structured and rigorous
* Is cognitively (thinking) structured as compared to the procedural structure of scripted testing. This structure comes from Charter, time boxing etc.
* Is highly teachable and manageable
* It is not a technique but it is an approach. What actions you perform next is governed by what you are doing currently

**How to do Exploratory Testing**

Exploratory test preparation goes through following 5 stages detailed below and it is also called session based test management (SBTM Cycle):

1. **Create a Bug Taxonomy (classification)**
   * Categorize common types of faults found in the past projects
   * Analyze the root cause analysis of the problems or faults
   * Find the risks and develop ideas to test the application.
2. **Test Charter**
   * Test Charter should suggest
     1. what to test
     2. how it can be tested
     3. What needs to be looked
   * Test ideas are the starting point of exploration testing
   * Test charter helps determine how the end user could use the system
3. **Time Box**
   * This method includes a pair of testers working together not less than 90 minutes
   * There should not be any interrupted time in those 90 minutes session
   * Timebox can be extended or reduced by 45 minutes
   * This session encourages testers to react on the response from the system and prepare for the correct outcome
4. **Review Results:**
   * Evaluation of the defects
   * Learning from the testing
   * Analysis of coverage areas
5. **Debriefing:**
   * Compilation of the output results
   * Compare the results with the charter
   * Check whether any additional testing is needed

For Example, during exploratory execution, the following needs to be done:

* The mission of testing should be very clear
* Keep notes on what needs to be tested, why it needs to be tested and the assessment of the product quality
* Tracking of questions and issues raised during exploratory testing
* Better to pair up the testers for effective testing
* The more we test, more likely to execute right test cases for the required scenarios

It is very important to take a document and monitor the following

* Test Coverage - Whether we have taken notes on the coverage of test cases and improve the quality of the software
* Risks - Which risks need to be covered and which are all important ones?
* Test Execution Log - Recordings on the test execution
* Issues / Queries - Take notes on the question and issues on the system

Smarter exploratory testing finds more errors in less time.

**Pros and Cons of Exploratory Testing**

|  |  |
| --- | --- |
| **Pro** | **Con** |
| * This testing is useful when requirement documents are not available or partially available * It involves Investigation process which helps find more bugs than normal testing- * Uncover bugs which are normally ignored by other testing techniques * Helps to expand the imagination of testers by executing more and more test cases which finally improves productivity as well * This testing drill down to the smallest part of an application and covers all the requirements * This testing covers all the types of testing and it covers various scenarios and cases * Encourages creativity and intuition * Generation of new ideas during test execution | * This testing purely depends on the tester skills * Limited by domain knowledge of the tester * Not suitable for Long execution time |

**Challenges of Exploratory Testing:**

There are many challenges of exploratory testing and those are explained below:

* Learning to use the application or software system is a challenge
* Replication of failure is difficult
* Determining whether tools need to be used can be challenging
* Determine the best test cases to execute can be difficult
* Reporting of the test results is a challenge as the report doesn't have planned scripts or cases to compare with the actual result or outcome
* Documentation of all events during execution is difficult to record
* Don't know when to stop the testing as exploratory testing has definite test cases to execute.

**When use exploratory testing?**

Exploratory testing can be used extensively when

* The testing team has experienced testers
* Early iteration is required
* There is a critical application
* New testers entered into the team

**Conclusion:**

In Software Engineering, Exploratory testing is performed to overcome the limitations of scripted testing. It helps in improving[Test Case](https://www.guru99.com/test-case.html)suite. It empathizes on learning and adaptability.

**hat is Mutation Testing?**

Mutation Testing is a type of software testing where we mutate (change) certain statements in the source code and check if the test cases are able to find the errors. It is a type of [White Box Testing](https://www.guru99.com/white-box-testing.html) which is mainly used for [Unit Testing](https://www.guru99.com/unit-testing-guide.html). The changes in mutant program are kept extremely small, so it does not affect the overall objective of the program.

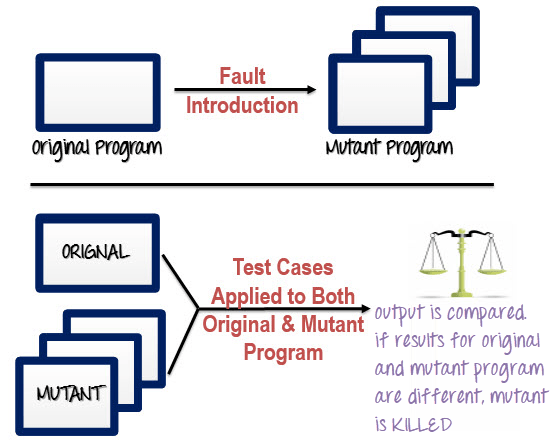
The goal of Mutation Testing is to assess the quality of the test cases which should be robust enough to fail mutant code. This method is also called as Fault-based testing strategy as it involves creating a fault in the program

Mutation was originally proposed in 1971 but lost fervor due to the high costs involved. Now, again it has picked steam and is widely used for languages such as[Java](https://www.guru99.com/java-tutorial.html)and XML.

In this tutorial, you will learn-

* [What is Mutation Testing?](https://www.guru99.com/mutation-testing.html#1)
* [How to execute Mutation Testing?](https://www.guru99.com/mutation-testing.html#2)
* [How to Create Mutant Programs?](https://www.guru99.com/mutation-testing.html#3)
* [What to change in a Mutant Program?](https://www.guru99.com/mutation-testing.html#4)
* [Types of Mutation Testing](https://www.guru99.com/mutation-testing.html#5)
* [Mutation Score:](https://www.guru99.com/mutation-testing.html#6)
* [Advantages of Mutation Testing:](https://www.guru99.com/mutation-testing.html#7)
* [Disadvantages of Mutation Testing:](https://www.guru99.com/mutation-testing.html#8)

**How to execute Mutation Testing?**

**[](https://www.guru99.com/images/m2.png)**

Following are the steps to execute mutation testing(mutation analysis):

**Step 1**: Faults are introduced into the source code of the program by creating many versions called mutants. Each mutant should contain a single fault, and the goal is to cause the mutant version to fail which demonstrates the effectiveness of the test cases.

**Step 2**: Test cases are applied to the original program and also to the mutant program. A[Test Case](https://www.guru99.com/test-case.html)should be adequate, and it is tweaked to detect faults in a program.

**Step 3**: Compare the results of an original and mutant program.

**Step 4**: If the original program and mutant programs generate the different output, then that the mutant is killed by the test case. Hence the test case is good enough to detect the change between the original and the mutant program.

**Step 5**: If the original program and mutant program generate the same output, Mutant is kept alive. In such cases, more effective test cases need to be created that kill all mutants.

**How to Create Mutant Programs?**

[](https://www.guru99.com/images/m3.png)

A mutation is nothing but a single syntactic change that is made to the program statement. Each mutant program should differ from the original program by one mutation.

| **Original Program** | **Mutant Program** |
| --- | --- |
| If (x>y)  Print "Hello"  Else  Print "Hi" | If(**x<y**)  **Print "Hello"**  **Else**  **Print "Hi"** |

**What to change in a Mutant Program?**

There are several techniques that could be used to generate mutant programs. Let's look at them

| **Operand replacement operators** | **Expression Modification Operators** | **Statement modification Operators** |
| --- | --- | --- |
| Replace the operand with another operand (x with y or y with x) or with the constant value. | Replace an operator or insertion of new operators in a program statement. | Programmatic statements are modified to create mutant programs. |
| Example-  If(x>y) replace x and y values  If(5>y) replace x by constant 5 | Example-  If(x==y)  We can replace == into >= and have mutant program as  If(x>=y) and inserting ++ in the statement  If(x==++y) | Example-  Delete the else part in an if-else statement  Delete the entire if-else statement to check how a program behaves  Some of sample mutation operators:   * GOTO label replacement * Return statement replacement * Statement deletion * Unary operator insertion (Like - and ++) * Logical connector replacement * Comparable array name replacement * Removing of else part in the if-else statement * Adding or replacement of operators * Statement replacement by changing the data * Data Modification for the variables * Modification of data types in the program |

**Automation of Mutation Testing:**

Mutation testing is extremely time consuming and complicated to execute manually. To speed up the process, it is advisable to go for automation tools. Automation tools reduce the cost of testing as well.

List of tools available -

* [Stryker](https://stryker-mutator.io/)
* [PIT Testing](http://pitest.org/)

**Types of Mutation Testing**

In Software Engineering, Mutation testing could be fundamentally categorized into 3 types– statement mutation, decision mutation, and value mutation.

1. **Statement Mutation**- developer cut and pastes a part of a code of which the outcome may be a removal of some lines
2. **Value Mutation**- values of primary parameters are modified
3. **Decision Mutation**- control statements are to be changed

**Mutation Score:**

The mutation score is defined as the percentage of killed mutants with the total number of mutants.

* Mutation Score = (Killed Mutants / Total number of Mutants) \* 100

[](https://www.guru99.com/images/m4.png)

Test cases are mutation adequate if the score is 100%. Experimental results have shown that mutation testing is an effective approach for measuring the adequacy of the test cases. But, the main drawback is that the high cost of generating the mutants and executing each test case against that mutant program.

**Advantages of Mutation Testing:**

Following are the advantages of Mutation Testing:

* It is a powerful approach to attain high coverage of the source program.
* This testing is capable comprehensively testing the mutant program.
* Mutation testing brings a good level of error detection to the software developer.
* This method uncovers ambiguities in the source code and has the capacity to detect all the faults in the program.
* Customers are benefited from this testing by getting a most reliable and stable system.

**Disadvantages of Mutation Testing:**

On the other side, the following are the disadvantages of Mutant testing:

* Mutation testing is extremely costly and time-consuming since there are many mutant programs that need to be generated.
* Since its time consuming, it's fair to say that this testing cannot be done without an automation tool.
* Each mutation will have the same number of test cases than that of the original program. So, a large number of mutant programs may need to be tested against the original test suite.
* As this method involves source code changes, it is not at all applicable for [Black Box Testing](https://www.guru99.com/black-box-testing.html).

**Conclusion:**

Do you want exhaustive testing of your application? The answer is Mutation testing. It is the most comprehensive technique to test a program. This is the method which checks for the effectiveness and accuracy of a testing program to detect the faults or errors in the system.

**What is Keyword Driven Framework?**

A keyword-driven framework is a table-driven testing or action word based testing. It is a software testing method suitable for both manual and automated testing.

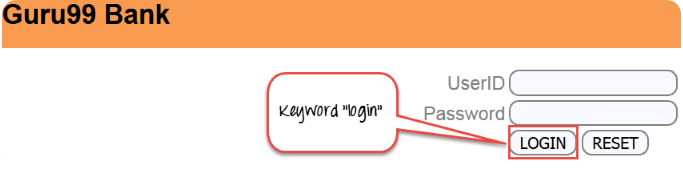
**What is Keyword Driven Testing?**

A keyword-driven testing is a scripting technique that uses data files to contain the keywords related to the application being tested. These keywords describe the set of actions that is required to perform a specific step.

A keyword-driven test consists of high and low-level keywords, including the keyword arguments, which is composed to describe the action of a test case.

In Keyword Driven Testing, you first identify a set of keywords and then associate an action (or function) related to these keywords. Here, every testing action like opening or closing of browser, mouse click, keystrokes, etc. is described by a keyword such as **openbrowser, click, Typtext**and so on.

**For Example**:

[](https://www.guru99.com/images/1/042616_0457_WhatisKeywo1.png)

1. **login** to "guru99" website – Keyword "login" will be used in our automation framework, to the test the login function or action associated with it.
2. **logout** to "guru99" website— Keyword "logout" will be used in our automation framework, to test the logout function or action associated with it.

We will see some more example further in the article.

**Example of keywords**

|  |  |
| --- | --- |
| Keywords | Description |
| Login | Login to guru99 bank demo site |
| Emails | Send Email |
| logouts | Log out from guru99 bank demo site |
| Notifications | Find unread notifications |

In order to create a Keyword driven framework, you need following things

1. **Excel Sheet**- Identify the keywords and store them in an Excel sheet
2. **Function Library**- Function library consist of the function for the business flows ( login button for any website).So when test is executed, it will read the keyword from the Excel sheet and call the functions accordingly
3. **Data Sheets**- Data sheets is used to store the test data that will be used in the application
4. **Object Repository**- based on your keyword driven framework you can use an object repository
5. **Test Scripts**- Based on the design of your framework, you can have test scripts for each manual[Test Case](https://www.guru99.com/test-case.html)or a single driver script

**Why do Keyword Driven Testing**

In Software Engineering, Keyword Driven Testing is done due to following reason

* Common components handled by standard library
* Using this approach tests can be written in a more abstract manner
* High degree of reusability
* The detail of the script is hidden from the users
* Users don't have to use the scripting languages
* The test is concise, maintainable and flexible

**How to perform Keyword Driven Testing**

Keyword based testing can be done in both ways, manually as well as automated. But usually, it is used with automated testing.

The objective behind automating Keyword based testing is

* It helps to reduce maintenance cost
* Avoids duplicated specifications
* Greater reuse of function scripting
* Better testing support and portability
* Achieve more testing with less or same effort

With keyword driven testing, you can create a simple functional tests in the earlier stages of development, testing the application piece-by-piece. The simplest way to compose keyword driven test is to record them. After recording, the test can be modified and customized as per the requirement.

Each keyword needs to be linked with atleast one command, test scripts or function, which implement the actions related to that keyword.

When test cases are executed, keywords are interpreted by a test library, which is called by a test automation framework.

The major activities involved in keyword driven testing are

* **Step 1**. Identifying low level as well as high-level keywords
* **Step 2**. Implementing the keywords as executable
* **Step 3**. Creating test cases
* **Step 4**. Creating the driver scripts
* **Step 5**. Executing the automation test scripts

**Tools used for Keyword Driven Testing**

Few tools which are extensively used for Keyword driven testing.

* [HP QTP](https://www.guru99.com/quick-test-professional-qtp-tutorial.html)
* [Selenium](https://www.guru99.com/selenium-tutorial.html)

**Advantages of Keyword Driven Testing**

* It allows functional testers to plan test automation before the application is ready
* Tests can be developed without programming knowledge
* It is not dependent on a specific programming language or tool
* Compatible with any automation tools available in the market

**Sample test cases**

1. TC\_01: Login to guru99 demo site, find out how many transactions is carried out today
2. TC\_02: Login to guru99 demo site, send an email to one of your customer and then logout
3. TC\_03: Login to guru99 demo site and check for any notification received

**Summary:**

* A keyword-driven testing is a scripting technique that uses data files to contain the keywords related to the application being tested
* A keyword-driven testing usually performed by automated testing.
* Tests can be developed without programming knowledge
* Tests are compatible with any automation tools available in the market

**What is Risk Based Testing?**

Risk based testing is prioritizing the feature's, modules and functions of the Application Under Test based on impact and likelihood of failures. It involves assessing the risk based on the complexity, business criticality, usage frequency, visible areas,[Defect](https://www.guru99.com/defect-management-process.html)prone areas, etc.

Risk is the occurrence of an uncertain event with a positive or negative effect on the measurable success criteria of a project. It could be events that have occurred in the past or current events or something that could happen in the future. These uncertain events can have an impact on the cost, business, technical and quality targets of a project.

Risks can be positive or negative.

* **Positive risks** are referred to as opportunities and help in business sustainability. For example investing in a New project, Changing business processes, Developing new products.
* **Negative Risks** are referred to as threats and recommendations to minimize or eliminate them must be implemented for project success.

In this tutorial, you will learn-

* [When to implement Risk based Testing](https://www.guru99.com/risk-based-testing.html#2)
* [Risk Management Process](https://www.guru99.com/risk-based-testing.html#3)
* [Risk Based Testing Approach](https://www.guru99.com/risk-based-testing.html#4)
* [Risk Based Testing Approach to the System Test](https://www.guru99.com/risk-based-testing.html#5)
* [How to do Risk Based Testing: complete Process](https://www.guru99.com/risk-based-testing.html#6)
* [Prioritization and Risk Assessment Matrix](https://www.guru99.com/risk-based-testing.html#7)
* [Generic Check list for Risk Based Testing](https://www.guru99.com/risk-based-testing.html#8)
* [Risk Based Testing Results Reporting and Metrics](https://www.guru99.com/risk-based-testing.html#9)
* [Inherent Risk vs. Residual Risk Assessment](https://www.guru99.com/risk-based-testing.html#10)
* [Benefits of Risk Based Testing](https://www.guru99.com/risk-based-testing.html#11)

**When to implement Risk based Testing**

Risk based testing can be implemented in

* Projects having time, resource, budget constraints, etc.
* Projects where risk based analysis can be used to detect vulnerabilities to SQL injection attacks.
* Security Testing in Cloud Computing Environments.
* New projects with high risk factors like Lack of experience with the technologies used, Lack of business domain knowledge.
* Incremental and iterative models, etc.

**Risk Management Process**

Let's now understand the steps involved in Risk Management Process

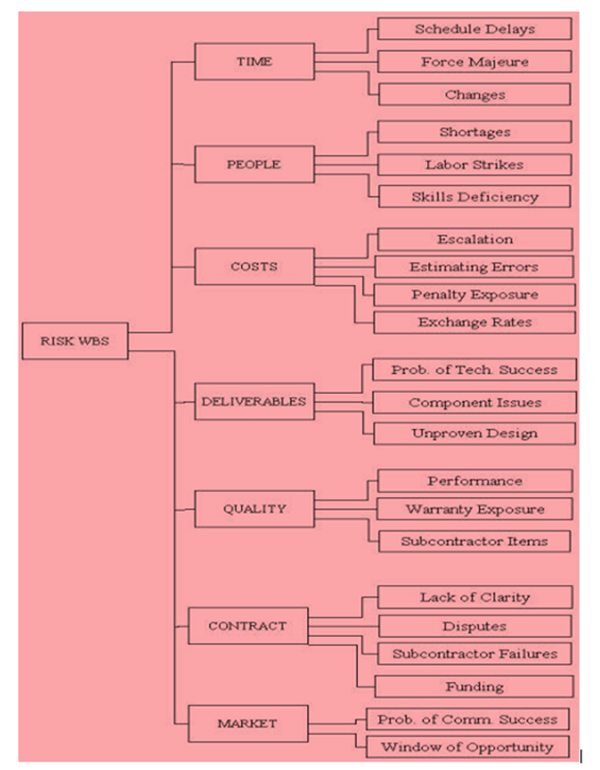
**Risk Identification**

Risk identification can be done through risk workshops, checklists, brainstorming, interviewing, Delphi technique, cause and effect diagrams, lessons learnt from previous projects, root cause analysis, contacting domain experts and subject matter experts.

**Risk Register** is a spreadsheet which has a list of identified risks, potential responses, and root causes. It is used to monitor and track the risks (both threats and opportunities) throughout the life of the project. Risk response strategies can be used to manage positive and negative risks.

Risk breakdown structure plays an important role in risk planning. The Risk Breakdown structure would help in identifying the risk prone areas and helps in effective evaluation and risk monitoring over the course of the project. It helps in providing sufficient time and resources for risk management activities. It also helps in categorizing many sources from which the project risks may arise.

**Risk Breakdown structure sample**

[](https://www.guru99.com/images/3-2016/032316_1114_RiskBasedTe3.png)

**Risk Analysis (Includes Quantitative and Qualitative Analysis)**

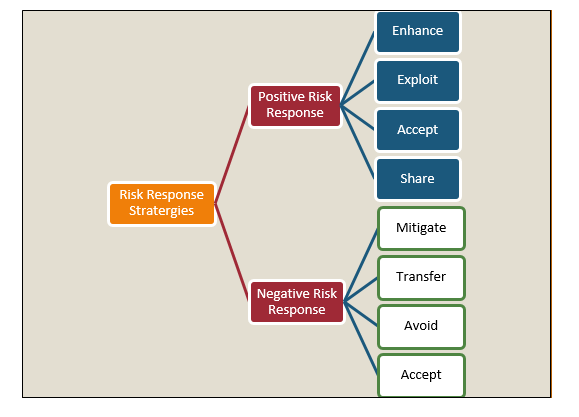
Once the list of potential risks has been identified, the next step is to analyze them and to filter the risk based on the significance. One of the qualitative risk analysis technique is using Risk Matrix (covered in the next section). This technique is used to determine the probability and impact of the risk.

**Risk Response planning**

Based on the analysis, we can decide if the risks require a response. For example, some risks will require a response in the project plan while some require a response in the project monitoring, and some will not require any response at all.

The risk owner is responsible for identifying options to reduce the probability and impact of the assigned risks.

**Risk mitigation** is a risk response method used to lessen the adverse impacts of possible threats. This can be done by eliminating the risks or reducing them to an acceptable level.

[](https://www.guru99.com/images/3-2016/032316_1114_RiskBasedTe4.png)

**Risk Contingency**

Contingency can be described as a possibility of an uncertain event, but the impact is unknown or unpredictable. A contingency plan is also known as the action plan/back up plans for the worst case scenarios. In other words, it determines what steps could be taken when an unpredictable event materializes.

**Risk Monitoring and Control**

Risk control and monitor process are used to track the identified risks, monitor residual risks, identify new risks, update the risk register, analyze the reasons for the change, execute risk response plan and monitor risk triggers, etc. Evaluate their effectiveness in reducing risks.

This can be achieved by risk reassessments, risk audits, variance and trend analysis, technical performance measurement, status update meetings and retrospective meetings.

The table below gives information about the

|  |  |  |
| --- | --- | --- |
| **Inputs to Risk Monitoring and Control** | **Tools and Techniques for Risk Monitoring and Control** | **Outputs from Risk Monitoring and Control** |
| Risk Management plan | Project risk response audits | Workaround plans |
| Risk Response Plan | Periodic project risk reviews | Corrective action |
| Project Communication plan | Earned value analysis | Project change requests |
| Additional Risk identification and analyis | Technical performance measurement | Updates to the riskResponse plan and risk Identification checklist |
| Scope changes | Additional risks response planning | Risk database |

We need to remember that risk increases with changes in technology, the size of the project, length of the project (Longer project timeframe), the number of sponsoring agencies, project estimates, efforts, and a shortage of appropriate skills.

**Risk Based Testing Approach**

1. Analyze the requirements.
2. Documents (SRS, FRS, Usecases) are reviewed. This activity is done to find and eliminate errors & ambiguities.
3. Requirements sign-off's is one of the risk-reduction technique for avoiding the introduction of late changes into the projects. Any changes to requirements after the document are baselined would involve a change control process and subsequent approvals.
4. Assess the risks by calculating the likelihood and impact each requirement could have on the project taking the defined criteria's like cost, schedule, resources, scope, technical performance safety, reliability, complexity, etc. into consideration.
5. Identify the probability of failure and high-risk areas. This can be done using risk assessment matrix.
6. Use a risk register to list the set of identified risks. Update, monitor and track the risks periodically at regular intervals.
7. Risk profiling needs to be done at this stage to understand the risk capacity and risk tolerance levels.
8. Prioritize the requirements based on the rating.
9. Risk-based test process is defined
10. Highly critical and medium risks can be considered for mitigation planning, implementation, progress monitoring. Low risks can be considered on a watch list.
11. Risk data quality assessment is done to analyze the quality of the data.
12. Plan and define test according to the rating
13. Apply appropriate testing approach and test design techniques to design the test cases in a way that the highest risks items are tested first. High-risk items can be tested by the resource with good domain knowledge experience.
14. Different test design techniques can be used for e.g. using the decision table technique on high-risk test items and using 'only' equivalence partitioning for low-risk test items.
15. Test cases are also designed to cover multiple functionalities and end to end business scenarios.
16. Prepare test data and test conditions and test bed.
17. Review the Test plans, Test Strategy, Test cases, Test reports or any other document created by the testing team.
18. Peer review is an important step in defect identification and risk reduction.
19. Perform dry runs and quality checks on the results
20. Test cases are executed according to the priority of the risk item.
21. Maintain traceability between risk items, tests that cover them, results of those tests, and defects found during testing. All testing strategies executed properly will reduce quality risks.
22. Risk-based testing can be used at every level of testing, e.g. component, integration, system, and acceptance testing
23. At the system level, we need to focus on what is most important in the application. This can be determined by looking at the visibility of functions, at frequency of use and at the possible cost of failure.
24. Evaluation of exit criteria. All high-risk areas fully tested, with only minor residual risks left outstanding.
25. Risk-based Test Results reporting and metrics analysis.
26. Reassess existing risk events and new risk events based on Key Risk Indicators.
27. Risk register updation.
28. Contingency plans- This works as a fallback plan/emergency plans for the high exposure risks.
29. Defect analysis and defect prevention to eliminate the defects.
30. Retesting and Regression testing to validate the defect fixes based on pre-calculated risk analysis and

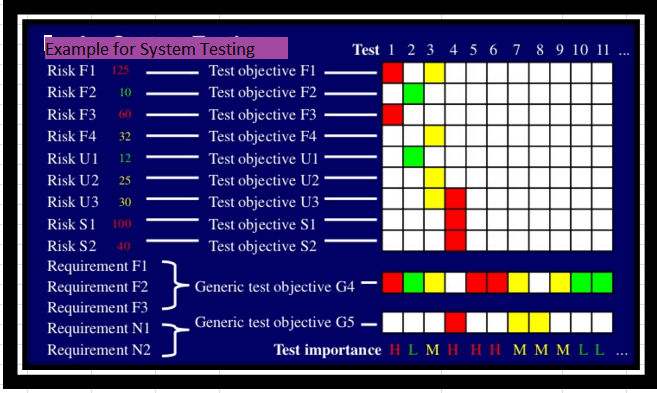
high-risk areas should be most intensively covered.

1. Risk-based automation testing(if feasible)
2. Residual Risk calculation
3. Risk Monitoring and Control
4. Exit Criteria or completion criteria can be used for different risk levels. All key risks have been addressed with appropriate actions or contingency plans. Risk exposure is at or below the level agreed to as acceptable for the project.
5. Risk profiling reassessment and customer feedback.

**Risk Based Testing Approach to the System Test**

1. **Technical System Test** –This is referred to as environment test and integration test. Environment test includes testing in development, testing, and the production environment.
2. **Functional System Test**- Testing of all functionalities, features, programs, modules. The purpose of this test is to evaluate if the system meets its specified requirements.
3. **Non-functional System Test**-Testing the non-functional requirements performance, load tests, stress-test, configuration tests, Security tests, backup and recovery procedures and documentation (system, operation and installation documentation).

Diagram below gives a clear overview of the above-mentioned process

[](https://www.guru99.com/images/3-2016/032316_1114_RiskBasedTe5.png)

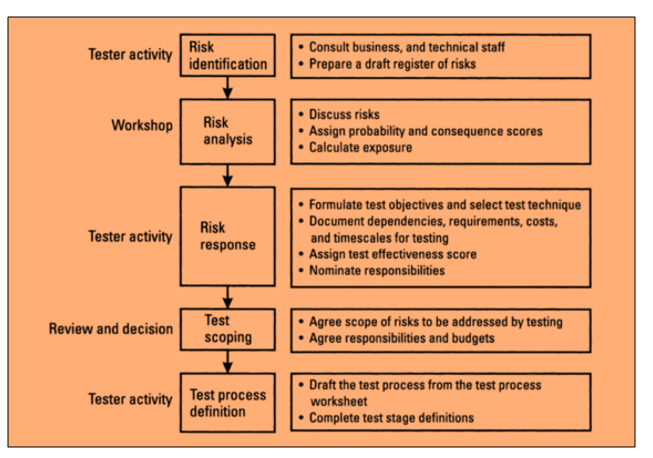
System Testing includes both Functional tests as well as Non-Functional tests.

Functional testing ensures that the product/application meets customer and business requirements. On the other hand, non-functional testing is done to verify if the product stands up to customer's expectations in terms of quality, reliability usability, performance, compatibility, etc.

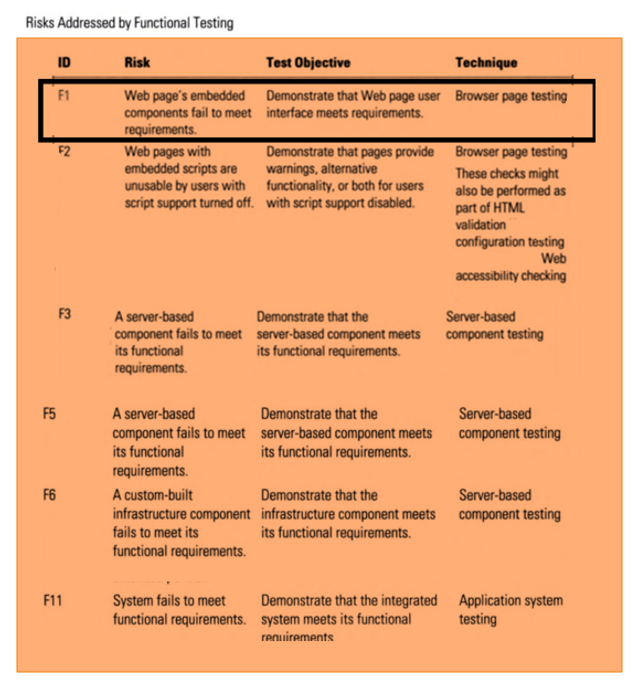
**How to do Risk Based Testing: Complete Process**

This section covers, Risk based Test Process

1. Risk Identification
2. Risk Analysis
3. Risk Response
4. Test Scoping
5. Test Process definition

[](https://www.guru99.com/images/3-2016/032316_1114_RiskBasedTe6.png)

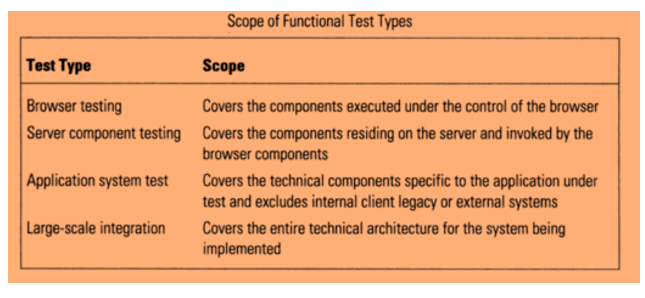
1. In this process, the risks are identified and categorized, a draft register of risks are prepared, risk sorting is done to identify the significant risks.
2. Risk response involves formulating the test objectives from the risks and selecting appropriate techniques to demonstrate the test activity /test technique to meet the test objectives.
3. Document dependencies, requirements, cost, time required for Software testing, etc. are considered to calculate the test effectiveness score.
4. Test scoping is a review activity that requires the participation of all stakeholders and technical staff. It is important to adhere to the agreed scope of risks. These risks need to be addressed by testing, and all members agree with the responsibilities assigned to them and budget allocated for these activities.
5. After the scope of testing has been finalized the test objectives, assumptions, dependencies for each test stages has to be compiled in the standard format.

[](https://www.guru99.com/images/3-2016/032316_1114_RiskBasedTe8.png)

Lets, consider the functional requirements F1, F2 ,F3 and Non-functional requirements N1 & N2

* **F1-Functional Requirement, R1-Risk Associated with F1**
  + Test Objective 1- Demonstrate using a Test that the expected features and functionalities of the system work fine, and the risk R1 can be addressed by functional testing
  + **Test**-Browser Page testing is done to execute important user tasks and verify that the R1 ( Risk associated with F1) could be addressed in a range of scenarios.
* **F2-Functional Requirement, R2-Risk Associated with F2**
  + Test Objective 2- Demonstrate using a **Test** that the expected features and functionalities of the system works fine, and the risk R2 can be addressed by functional testing
  + **Test**-Browser Page testing is done to execute important user tasks and verify that the R2 could be addressed in a range of scenarios
* **F3-Functional Requirement, R3-Risk Associated with F3**
  + Test Objective 3- Demonstrate using a **Test** that the expected features and functionalities of the system works fine, and the risk R3 can be addressed by functional testing
  + **Test**-Browser Page testing is done to execute important user tasks and verify that R3 could be addressed in a range of scenarios
* **N1- Non -Functional Requirement, NR1-Risk Associated with N1**
  + Test Objective N1-Demonstrate using a **Test** that the operational characteristics of the system works fine and the risk NR1 can be addressed by non-functional testing
  + **Test**-Usability testing is a technique used to assess how easy user interfaces are to use and verify that the NR1 could be addressed by usability testing
* **N2- Non -Functional Requirement, NR2-Risk Associated with N2**
  + Test Objective N.2- Demonstrate using a Test that the operational characteristics of the system works fine, and the risk NR2 can be addressed by non-functional testing
  + Test-Security testing is a technique used to check whether the application secured or is it vulnerable to attacks, whether there is any information leakage and verifies that NR2 could be addressed by security testing.

**Specific Test objectives**: The risks and test objectives listed are specific to the test types.

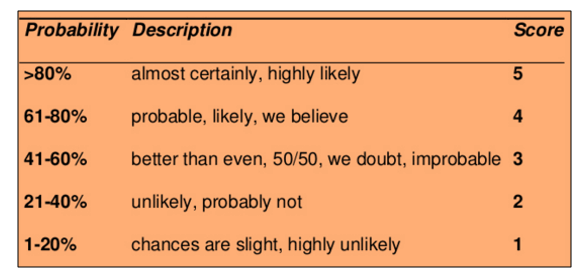
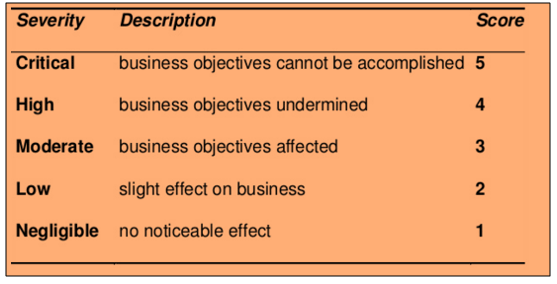
[](https://www.guru99.com/images/3-2016/032316_1114_RiskBasedTe9.png)

**Procedure to design the risk based test process**

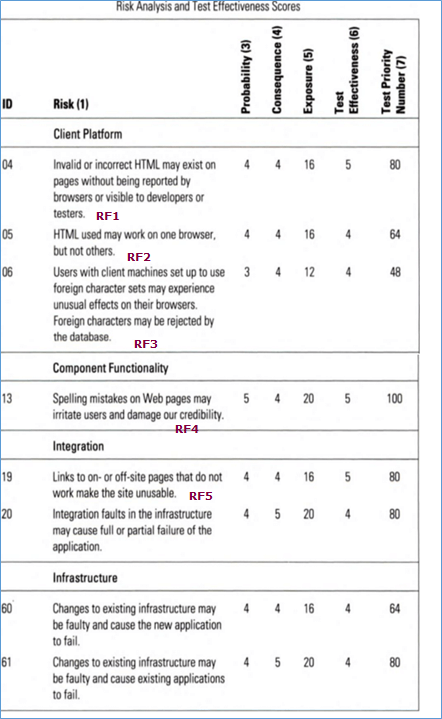
* Prepare a risk register.This records the risks derived from generic risk list, existing checklist, brainstorming session.
* Include the risks associated with the system functional and non-functional requirements (Usability, security,performance)
* Each risk is allocated a unique identifier

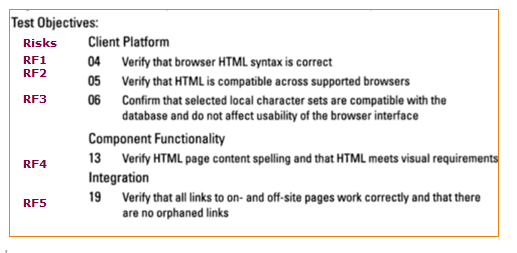
|  |  |  |
| --- | --- | --- |
| **Col No.** | **Column Heading** | **Description** |
| 3 | Probability | Likelihood of the system prone to this mode of failure |
| 4 | Consequences | impact of this mode of failure |
| 5 | Exposure | Product of Probability and Consequences (columm 3&4) |
| 6 | Test effectiveness | How confident are the testers that they can address this risk? |
| 7 | Test priority number | Product of Probability, Consequences and Test effectiveness (column 3,4 6) |
| 8 | Test objective(s) | what test objective will be used to address this risk |
| 9 | Test techniques | what method or technique is used to |
| 10 | Dependencies | What do the testers assume and depend on |
| 11 | Effort | How much effort is required to this testing |
| 12 | Timescale | How much of time is required to do this testing |
| 13 | Test stage A-Unit TestsTest stage B-Integration TestTest Stage C-System Test | Name of the person or group doing this activity |

The Probability(1 Low -5 High ) and consequences(1 Low -5 High ) of each risk are assessed

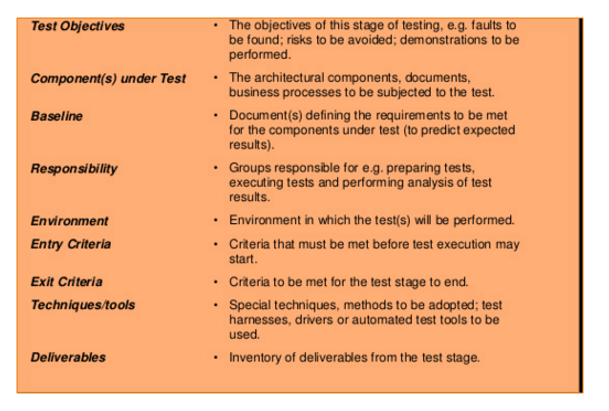
[](https://www.guru99.com/images/3-2016/032316_1114_RiskBasedTe10.png) [](https://www.guru99.com/images/3-2016/032316_1114_RiskBasedTe11.png)

* Test exposure is computed
* The tester analyzes each risk and evaluates whether the risk is testable or not
* Test objectives are defined for the testable risks
* Tester specifies the test activity that should be carried out in a planned way to meet the test objective(Static reviews, inspections, sytem tests, integration tests, acceptance tests, html validation, localization testing, etc.,)
* These testing activities can be classified into stages (Component Testing/Unit testing, Integration Testing, System Testing, Acceptance Testing)
* At times, a risk might be addressed by one or more than one test stage
* Identify the dependencies and assumptions (Availability of skills, tools, test environments, resources)
* Test effectiveness is computed. Test effectiveness relates to the confidence level of the tester that the risk will be definitively addressed through testing. Test effectiveness score is a number between one and five.( 5-High Confidence , 1-Low Confidence)
* Estimate of the effort, the time required, cost to prepare and execute these tests.

[](https://www.guru99.com/images/3-2016/032316_1114_RiskBasedTe12.png)

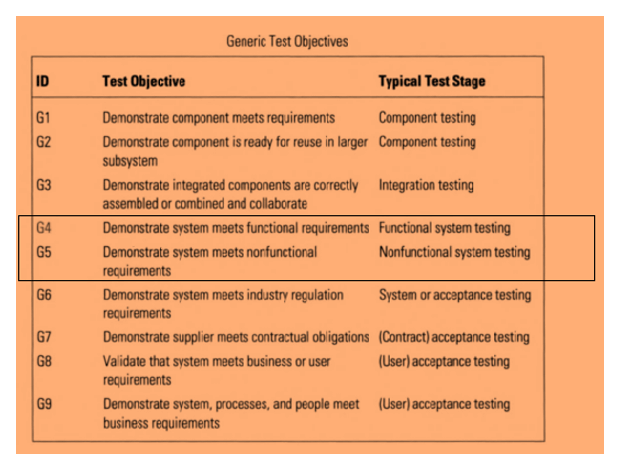
[](https://www.guru99.com/images/3-2016/032316_1114_RiskBasedTe13.png)

* Test priority number is calculated. It is the product of probability, consequences, and test effectiveness scores.
  + 125-MaximumàA very serious risk that could be detected with testing
  + 1-Minimum àA very low risk that would not be detected with testing
* Based on the test priority number the test importance can be classified as High(Red ), Medium (Yellow) &Low (Green). Highest risk items are tested first.
* Allocation the test activities to the test stages.Designate the group that will perform testing for each objective in the different test stages(Unit testing, Integration Testing, System Testing, Acceptance Testing)
* What is in scope and out of scope for testing is decided in the test scoping phase
* For each stage test objectives, component under test, responsibility,environment,entry criteria,exit criteria,tools,techniques,deliverables are defined.

[](https://www.guru99.com/images/3-2016/032316_1114_RiskBasedTe14.png)

Generic Test Objectives- These generic objectives are applicable to multiple projects and applications

* Component meets the requirement and is ready for use in larger subsystems
* The risks associated with the specific test types are addressed, and the test objectives are accomplished.
* Integrated components are correctly assembled. Ensure interface compatibility among the components.
* The system meets the specified functional and nonfunctional requirements.
* Product components satisfy end user needs in their intended operating environment
* Risk management strategy is used to identifying, analyzing, and mitigating risks.
* The System meets industry regulation requirements
* The System meets contractual obligations
* Institutionalization and the achievement of other specific objectives established such as cost, schedule, and quality objectives.
* System, processes and people meet business requirements

[](https://www.guru99.com/images/3-2016/032316_1114_RiskBasedTe15.png)

Generic Test objectives can be defined for the different test stages

* Component Testing
* Integration Testing
* System Testing
* Acceptance Testing

Let's consider the system test stage

1. G4 & G5 demonstrates's the system meets the functional (F1,F2,F3) and non-functional requirements(N1,N2) .
2. Demonstrate using tests that the expected features and functionalities of the system work fine and the risk associated with F1, F2, F3 can be addressed by functional testing
3. Demonstrate using tests that the operational characteristics of the system work fine and the risk associated with N1, N2 can be addressed by non-functional testing
4. Based on the test priority number the test importance can be classified as High(Red ), Medium (Yellow) &Low (Green).

**Prioritization and Risk Assessment Matrix**

Risk assessment matrix is the probability impact matrix. It provides the project team with a quick view of the risks and the priority with which each of these risks needs to be addressed.

Risk rating = Probability x Severity

Probability is the measure of the chance for an uncertain event will occur. Exposure in terms of time, proximity and repetition. It is expressed in terms of percentage.

This can be classified as Frequent(A), Probable(B), Occasional(C), Remote(D), Improbable(E), Eliminated(F)

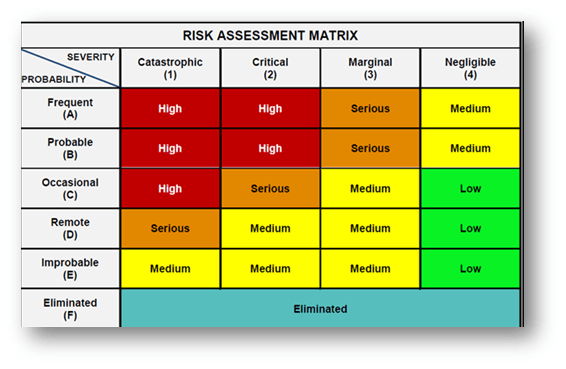
* Frequent- It is expected to occur several times in most circumstances (91 - 100%)
* Probable: Likely to occur several times in most circumstances (61 - 90%)
* Occasional: Might occur sometime (41 - 60%)
* Remote –Unlikely to occur /could occur sometime ( 11 - 40%)
* Improbable-May occur in rare and exceptional circumstances (0 -10%)
* Eliminate-Impossible to occur (0%)

Severity is the degree of impact of damage or loss caused due to the uncertain event. Scored 1 to 4 and can be classified as Catastrophic=1, Critical=2, Marginal=3, Negligible=4

* **Catastrophic**– Harsh Consequences that make the project completely unproductive and could even lead to project shutdown. This must be a top priority during risk management.
* **Critical**- Large consequences which can lead to a great amount of loss. Project is severely threatened.
* **Marginal**– Short term damage still reversible through restoration activities.
* **Negligible**- Little or minimal damage or loss. This can be monitored and managed by routine procedures.

The priority is classified into four categories, which is mapped against the severity and probability of the risk as shown in below image.

* Serious
* High
* Medium
* Low

[](https://www.guru99.com/images/3-2016/032316_1114_RiskBasedTe16.png)

**Serious:** The risks that fall in this category are marked in Amber color. The activity must be stopped, and immediate action must be taken to isolate the risk. Effective controls must be identified and implemented. Further, the activity must not proceed unless the risk is reduced to a low or medium level.

**High:** The risks that fall in this category are marked in Red color ate action or risk management strategies. Immediate action must be taken to isolate, eliminate, substitute the risk and to implement effective risk controls. If these issues cannot be resolved immediately, strict timelines must be defined to resolve these issues.

**Medium:** The risks that fall in this category are marked in Yellow color. Reasonable and practical steps must be taken to minimize the risks.

**Low:** The risks that fall in this category are marked in green color) marked can be ignored as they usually do not pose any significant problem. Periodical review is a must to ensure the controls remain effective

**Generic Check list for Risk Based Testing**

Comprehensive list of important points to be considered in Risk based testing

* Important functionalities in the project.
* User visible functionality in the project
* The functionality having the largest safety impact
* Functionalities that have largest financial impact on users
* Highly Complex areas of source code and error prone codes
* Features or functions that can be tested early in the development cycle.
* Features or functionalities were added to the product design in the last minute.
* Critical factors of similar/related previous projects that caused problems/issues.
* Prime factors or issues of Similar/related projects that that had a huge impact on the operation and maintenance expenses.
* Poor requirements that lead to poor designs and tests that could have an impact on the project goals and deliverables.
* In the worst case scenario, a product may be so defective that it cannot be reworked and must be completely scrapped this would cause serious harm to the company's reputation. Identify what kind of problems are crucial to the product objectives.
* Situations or problems that would cause persistent customer service complaints.
* End to end Tests could easily focus on the multiple functionalities of the system.
* Optimal set of tests that can maximize the risk coverage
* Which tests will have the best high-risk-coverage to time-required ratio?

**Risk Based Testing Results Reporting and Metrics**

1. **Test report preparation**

Reporting test status is about effectively communicating the test results to the project stakeholders. And to give a clear understanding and to show the comparison of test results with test objectives.

* Number of test cases planned vs. executed
* Number of test cases passed/failed
* Number of defects identified and their Status & Severity
* Number of defects and their status
* Number of critical defects- still open
* Environment downtimes – if any
* Showstoppers – if any

Test Summary Report, Test coverage report

1. **Metrics Preparation**

Metrics is a combination of two or more measures used to compare software processes, projects, and products.

* + Effort and Schedule variation
  + Test Case Preparation Productivity
  + Test Design Coverage
  + Test case execution productivity
  + **Risk Identification Efficiency %**
  + **Risk Mitigation Efficiency %**
  + Test Effectiveness %
  + Test Execution Coverage
  + Test Execution productivity
  + Defect Leakage %
  + Defect detection efficiency
  + Requirement Stability Index
  + Cost of Quality

1. Analyze the risks in nonfunctional categories (performance, reliability, and usability) based on defect status and a number of test pass/fail status, based on their relationship to risks.
2. Analyze the risks in functional categories metrics of testing, defect status and test pass/fail status, based on their relationship to risks.
3. Identify key lead and lag indicators and create early warning indicators
4. Monitor and report on lead and lag risk indicators (Key Risk Indicators) by analyzing the data patterns, trends, and interdependencies.

**Inherent Risk vs. Residual Risk Assessment**

Risk identification and analysis should also include inherent risks, residual risks, secondary risks and recurrent risk

* **Inherent Risk**: The risks that were identified/already present in the system before the controls and responses were implemented. Inherent risks are also known as Gross risks
* **Residual Risk**: The risks that are left over after the controls and responses have been implemented. Residual risks are known as the net risks
* **Secondary Risk:**The new risk caused by the implementation of risk response plan
* **Recurrent risks:** Likelihood that the initial risks will occur.

Test result measurement based on risk helps the organization to know the residual level of quality risk during test execution, and to make smart release decisions.

**Risk Profiling and Customer Feedback**

Risk profiling is a process for finding the optimal level of investment risk for the client considering the risk required, risk capacity and risk tolerance.

1. Risk Required is the level of risk the client needs to take in order to obtain a satisfactory return
2. Risk capacity is the level of financial risk the client can afford to take
3. Risk tolerance is the level of risk which the client would prefer to take

**Customer Feedback**

Gather customer feedback and reviews to improve the business, product, service and experience.

**Benefits of Risk Based Testing**

The benefits of Risk Based Testing is given below

* Improved productivity and cost reduction
* Improved Market opportunity (Time to market) and On time delivery.
* Improved service performance
* Improved quality as all of the critical functions of the application are tested.
* Gives clear information on test coverage. Using this approach, we know what has/have not been tested.
* Test effort allocation based on risk assessment is the most efficient and effective way to minimize the residual risk upon release.
* Test result measurement based on risk analysis enables the organization to identify the residual level of quality risk during test execution, and to make smart release decisions.
* Optimized testing with highly defined risk evaluation methods.
* Improved customer satisfaction – Due to customer involvement and good reporting and progress tracking.
* Early detection of the potential problem areas. Effective preventive measures can be taken to overcome these problems
* Continuous risk monitoring and assessment throughout the project's entire lifecycle helps in identification and resolution of risks and address the issues that could endanger the achievement of overall project goals and objectives.

**Summary:**

In Software Engineering, Risk based testing is the most efficient way to guide the project based on risks.

The testing efforts are effectively organized, and level of priority of each risk item is rated. Each risk is then associated with the appropriate test activities, where a single test having more than one risk item, then the test is assigned as the highest risk.

Tests are executed according to the risk priority order. Risk monitoring process helps in keeping track of the identified risks, and reducing the impacts of residual risks.

**What is Backend Testing?**

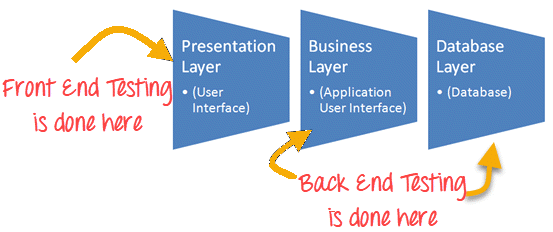
Backend testing is defined as a type of testing that checks the server side or Database. It is also known as Database Testing. The data entered in the front end will be stored in the back-end database. The database may be[SQL](https://www.guru99.com/sql.html)Server, MySQL, Oracle, DB2, etc. The data will be organized in the tables as record, and it is used to support the content of the page.

Database or backend testing is important because if it is not done it has some serious complications like deadlock, data corruption, data loss, etc.

**How to do Backend Testing**

Database testing mainly includes validating

* Schema
* Database tables
* Columns
* Keys and Indexes
* Stored procedures
* Triggers
* Database server validations
* Validating data duplication

[](https://www.guru99.com/images/1/111517_1154_FrontendTes1.png)

In back end testing, you are not required to use the GUI; you can directly pass the request through some browser with the parameters required for the function and get a response in some default format. Example, XML or JSON. You also need to connect to the database directly and verify the data using SQL queries. Through log files, debugging can be done.

There are various phases in back-end testing. The first step is to acquire design specification for a database server. The next step is to test specification design, followed by implementing the tests in this design with SQL code.

**Types of database testing**

The types of database testing includes

* Structural Testing
* Functional Testing
* Non-Functional Testing

**Tools used for database testing**

Some of the useful tools used for database testing includes

* [Data Factory](https://sourceforge.net/projects/data-factory/)
* [Data Generator](http://www.sqledit.com/dg/download.html)

**Advantages for back end testing**

* Back end testing is not like a [Black Box Testing](https://www.guru99.com/black-box-testing.html)
* Full control of[Test coverage](https://www.guru99.com/test-coverage-in-software-testing.html)and depth
* In early development stage, many bugs can be effectively found

In order to do the back end testing, tester is expected to have strong background in database server and knowledge of structured query language.

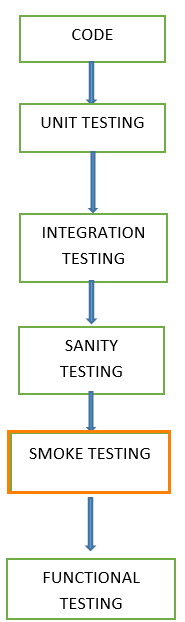
**What is Smoke Testing?**

Smoke testing is defined as a type of software testing that determines whether the deployed build is stable or not. This serves as confirmation whether the QA team can proceed with further testing. Smoke tests are a minimal set of tests run on each build. Here is the cycle where smoke testing is involved

Smoke testing is a process where the software build is deployed to QA environment and is verified to ensure the stability of the application. It is also called as "Build verification Testing" or “Confidence Testing.”

In simple terms, we are verifying whether the important features are working and there are no showstoppers in the build that is under testing.

It is a mini and rapid regression test of major functionality. It is a simple test that shows the product is ready for testing. This helps determine if the build is flawed as to make any further testing a waste of time and resources.

[](https://www.guru99.com/images/3-2016/032816_1308_SmokeTestin1.png)

Learn about comparison [Smoke Vs Sanity Testing](https://www.guru99.com/smoke-sanity-testing.html)

The smoke tests qualify the build for further formal testing. The main aim of smoke testing is to detect early major issues. Smoke tests are designed to demonstrate system stability and conformance to requirements.

A build includes all data files, libraries, reusable modules, engineered components that are required to implement one or more product functions.

In this tutorial, you will learn-

* [What is Smoke Testing?](https://www.guru99.com/smoke-testing.html#1)
* [When do we do smoke testing](https://www.guru99.com/smoke-testing.html#2)
* [Who will do Smoke Testing](https://www.guru99.com/smoke-testing.html#3)
* [Why do we do smoke testing?](https://www.guru99.com/smoke-testing.html#4)
* [How to do Smoke Testing?](https://www.guru99.com/smoke-testing.html#5)
* [Advantages of Smoke testing](https://www.guru99.com/smoke-testing.html#6)
* [Sample Smoke Test Cases Example](https://www.guru99.com/smoke-testing.html#7)

**When do we do smoke testing**

Smoke Testing is done whenever the new functionalities of software are developed and integrated with existing build that is deployed in QA/staging environment. It ensures that all critical functionalities are working correctly or not.

In this testing method, the development team deploys the build in QA. The subsets of test cases are taken, and then testers run test cases on the build. The QA team test the application against the critical functionalities. These series of test cases are designed to expose errors that are in build. If these tests are passed, QA team continues with[Functional Testing](https://www.guru99.com/functional-testing.html).

Any failure indicates a need to handle the system back to the development team. Whenever there is a change in the build, we perform Smoke Testing to ensure the stability.

**Example**: -New registration button is added in the login window and build is deployed with the new code. We perform smoke testing on a new build.

**Who will do Smoke Testing**

After releasing the build to QA environment, Smoke Testing is performed by QA engineers/QA lead. Whenever there is a new build, QA team determines the major functionality in the application to perform smoke testing. QA team checks for showstoppers in the application that is under testing.

Testing done in a development environment on the code to ensure the correctness of the application before releasing build to QA, this is known as Sanity testing. It is usually narrow and deep testing. It is a process which verifies that the application under development meets its basic functional requirements.

Sanity testing determines the completion of the development phase and makes a decision whether to pass or not to pass software product for further testing phase.

**Why do we do smoke testing?**

Smoke testing plays an important role in software development as it ensures the correctness of the system in initial stages. By this, we can save test effort. As a result, smoke tests bring the system to a good state. Once we complete smoke testing then only we start functional testing.

* All the show stoppers in the build will get identified by performing smoke testing.
* Smoke testing is done after the build is released to QA. With the help of smoke testing, most of the defects are identified at initial stages of software development.
* With smoke testing, we simplify the detection and correction of major defects.
* By smoke testing, QA team can find defects to the application functionality that may have surfaced by the new code.
* Smoke testing finds the major severity defects.

**Example 1:**Logging window: Able to move to next window with valid username and password on clicking submit button.

**Example 2:**Userunable to sign out from the webpage.

**How to do Smoke Testing ?**

Smoke Testing is usually done manually though there is a possibility of accomplishing the same through automation. It may vary from organization to organization.

**Manual Smoke testing**

In general, smoke testing is done manually. It approaches varies from one organization to other. Smoke testing is carried to ensure the navigation of critical paths is as expected and doesn't hamper the functionality. Once the build is released to QA, high priority functionality test cases are to be taken and are tested to find the critical defects in the system. If the test passes, we continue the functional testing. If the test fails, the build is rejected and sent back to the development team for correction. QA again starts smoke testing with a new build version. Smoke testing is performed on new build and will get integrated with old builds to maintain the correctness of the system. Before performing smoke testing, QA team should check for correct build versions.

**Smoke testing by Automation**

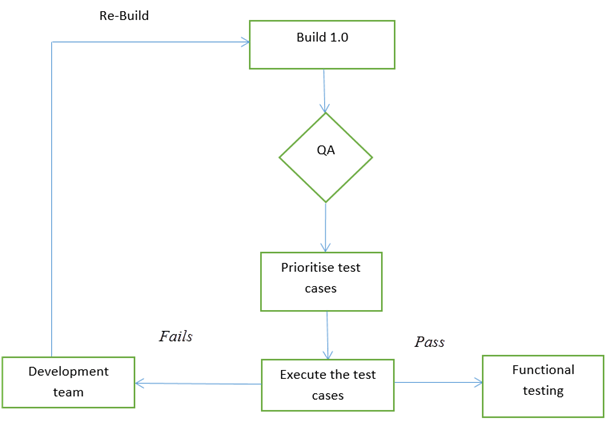
[Automation Testing](https://www.guru99.com/automation-testing.html) is used for [Regression Testing](https://www.guru99.com/regression-testing.html). However, we can also use a set of automated test cases to run against Smoke Test. With the help of automation tests, developers can check build immediately, whenever there is a new build ready for deployment.

Instead of having repeated test manually whenever the new software build is deployed, recorded smoke test cases are executed against the build. It verifies whether the major functionalities still operates properly. If the test fails, then they can correct the build and redeploy the build immediately. By this, we can save time and ensure a quality build to the QA environment.

Using an automated tool, test engineer records all manual steps that are performed in the software build.

**Smoke testing cycle**

Below flow chart shows how Smoke Testing is executed. Once the build is deployed in QA and, smoke tests are passed we proceed for functional testing. If the smoke test fails, we exit testing until the issue in the build is fixed.

[](https://www.guru99.com/images/3-2016/032816_1308_SmokeTestin2.png)

Smoke test cycle

**Advantages of Smoke testing**

Here are few advantages listed for Smoke Testing.

* Easy to perform testing
* Defects will be identified in early stages.
* Improves the quality of the system
* Reduces the risk
* Progress is easier to access.
* Saves test effort and time
* Easy to detect critical errors and correction of errors.
* It runs quickly
* Minimises integration risks

**What happens if we don't do Smoke testing**

If we don't perform smoke testing in early stages, defects may be encountered in later stages where it can be cost effective. And the[Defect](https://www.guru99.com/defect-management-process.html)found in later stages can be show stoppers where it may affect the release of deliverables.

**Sample Smoke Test Cases Example**

| **T.ID** | **TEST SCENARIOS** | **DESCRIPTION** | **TEST STEP** | **EXPECTED RESULT** | **ACTUAL RESULT** | **STATUS** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Valid login credentials | Test the login functionality of the web application to ensure that a registered user is allowed to login with username and password | 1.Launch the application 2.Navigate the login page 3.Enter valid username 4.Enter valid password 5.Click on login button | Login should be success | as expected | Pass |
| 2 | Adding item functionality | Able to add item to the cart | 1.Select categories list 2.Add the item to cart | Item should get added to the cart | Item is not getting added to the cart | Fail |
| 3 | Sign out functionality | Check sign out functionality | 1. select sign out button | The user should be able to sign out. | User is not able to sign out | Fail |

**Summary:**

In Software Engineering, Smoke testing should be performed on each and every build without fail as it helps to find defects in early stages. Smoke test activity is the final step before the software build enters the system stage. Smoke tests must be performed on each build that is turned to testing. This applies to new development and major and minor releases of the system.

Before performing smoke testing, QA team must ensure the correct build version of the application under test. It is a simple process which takes a minimum time to test the stability of the application.

Smoke tests can minimise test effort, and can improve the quality of the application. Smoke testing can be done either manually or by automation depending on the client and the organization.

**What is Accessibility Testing?**

Accessibility Testing is defined as a type of Software Testing performed to ensure that the application being tested is usable by people with disabilities like hearing, color blindness, old age and other disadvantaged groups. It is a subset of [Usability Testing](https://www.guru99.com/usability-testing-tutorial.html).

People with disabilities use assistive technology which helps them in operating a software product. Examples of such software are:

* **Speech RecognitionSoftware -** It will convert the spoken word to text , which serves as input to the computer.
* **Screen reader software** - Used to read out the text that is displayed on the screen
* **Screen Magnification Software**- Used to enlarge the monitor and make reading easy for vision-impaired users.
* **Special keyboard** made for the users for easy typing who have motor control difficulties

[](https://www.guru99.com/images/Accessibility-testing.jpg)

In this tutorial, you will learn-

* [What is accessibility testing?](https://www.guru99.com/accessibility-testing.html#1)
* [Why accessibility Testing?](https://www.guru99.com/accessibility-testing.html#2)
* [Which disabilities to support?](https://www.guru99.com/accessibility-testing.html#3)
* [How to do accessibility testing?](https://www.guru99.com/accessibility-testing.html#4)
* [Accessibility Testing Tools](https://www.guru99.com/accessibility-testing.html#5)
* [Myths of Accessibility Testing](https://www.guru99.com/accessibility-testing.html#6)

**Why Accessibility Testing?**

**Reason 1**: Cater to market for Disabled People.

[](https://www.guru99.com/images/Cater-to-Market-For-disabled.jpg)

About 20% of the population has disability issues.

* 1 in 10 people have a sever disability
* 1 in 2 people over 65 have reduced capabilities

Disabilities include blindness, deaf, handicapped, or any disorders in the body.

A software product can cater to this big market, if it's made disabled friendly. Accessibility issues in software can be resolved if Accessibility Testing is made part of normal software testing life cycle.

**Reason 2**: Abide by Accessibility Legislations

[](https://www.guru99.com/images/Abide_by_Accessibility_Legislations(1).jpg)

Government agencies all over the world have come out with legalizations, which requires that IT products to be accessible by disabled people.

Following are the legal acts by various governments -

* United States: Americans with Disabilities Act - 1990
* United Kingdom: Disability Discrimination Act - 1995
* Australia: Disability Discrimination Act - 1992
* Ireland : Disability Act of 2005

Accessibility Testing is important to ensure legal compliance.

**Reason 3**: Avoid Potential Law Suits

[](https://www.guru99.com/images/Avoid_Potential_Law_Suits.jpg)

In the past,Fortune 500 companies have been sued because their products were not disabled friendly. Here a few prominent cases

* National Federation for the Blind (NFB) vs Amazon (2007)
* Sexton and NFB vs Target (2007)
* NFB Vs AOL settlement (1999)

It's best to create products which support disabled and avoid potential lawsuits.

**Which Disabilities to Support?**

Application must support people with disabilities like -

|  |  |
| --- | --- |
| **Type of Disability** | **Disability Description** |
| **Vision Disability** | * Complete Blindness or Color Blindness or Poor Vision * Visual problems like visual strobe and flashing effect problems |
| **Physical Disability** | * Not able to use the mouse or keyboard with one hand. * Poor motor skills like hand movements and muscle slowness |
| **Cognitive disability** | * Learning Difficulties or Poor Memory or not able to understand more complex scenarios |
| **Literacy Disability** | * Reading Problems |
| **Hearing Disability** | * Auditory problems like deafness and hearing impairments * Cannot able to hear or not able to hear clearly |

**How to do Accessibility Testing?**

Accessibility Testing can be performed in 2 ways, and they are:

1. Manual
2. Automated

Following are the point's needs to be checked for application to be used by all users. This checklist is used for signing off accessibility testing.

1. Whether an application provides keyboard equivalents for all mouse operations and windows?
2. Whether instructions are provided as a part of user documentation or manual? Is it easy to understand and operate the application using the documentation?
3. Whether tabs are ordered logically to ensure smooth navigation?
4. Whether shortcut keys are provided for menus?
5. Whether application supports all operating systems?
6. Whether response time of each screen or page is clearly mentioned so that End Users know how long to wait?
7. Whether all labels are written correctly in the application?
8. Whether color of the application is flexible for all users?
9. Whether images or icons are used appropriately, so it's easily understood by the end users?
10. Whether an application has audio alerts?
11. Whether a user is able to adjust audio or video controls?
12. Whether a user can override default fonts for printing and text displays?
13. Whether user can adjust or disable flashing, rotating or moving displays?
14. Check to ensure that color-coding is never used as the only means of conveying information or indicating an action
15. Whether highlighting is viewable with inverted colors? Testing of color in the application by changing the contrast ratio
16. Whether audio and video related content are properly heard by the disability people ? Test all multimedia pages with no speakers in websites
17. Whether training is provided for users with disabilities that will enable them to become familiar with the software or application?

Accessibility testing may be challenging for testers because they are unfamiliar with disabilities. It is better to work with disabled people who have specific needs to understand their challenges.

There are different way of Testing the Accessibility depending upon the Disability. We will learn all them one by one.

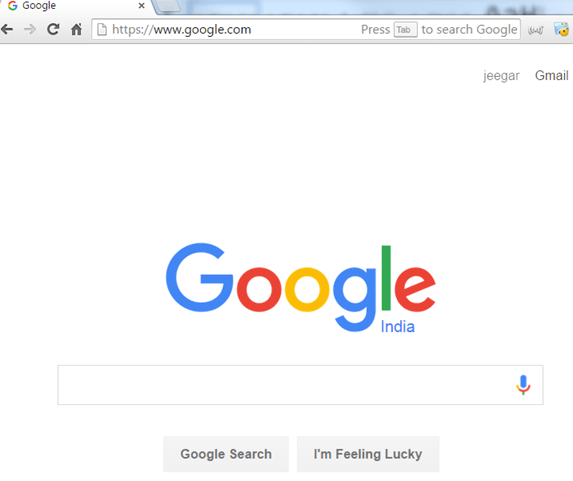
**1) Vision Disability**

OK now let us assume I don't have vision ability. I am completely blind, and I wanted to access XYZ Website. In that case, what is the option???? Cannot I access the XYZ website? What the option do I have? There is one-word option which is termed as **SCREENREADER.** Yeah, you got it right. SCREENREADER. Now, what is this Screen reader? It is a Software which is used for narrating the content on the web. Basically, what is on your website whether it is content, Link, Radio Button, Images, Video, etc. A screen reader will narrate each and everything for me. There are numerous Screen Reader available. I have worked with jaws.

Basically, when you start jaws or any screen reader and then go to the website, then it will narrate you the complete content. For Ex: I have started jaws, and started the browser JAWS will announce that Mozilla Firefox starts page, now if I go to address bar then JAWS will announce that **ADDRESS BAR** and then type **www.google.com** on address bar, jaws will going to explain somewhat like this:-

Address Bar,w,w,w,period,g,o,o,g,l,e,period,c,o,m. Also, when the page loads completely jaws will again announce Google.Com Home page.

Now if I go to Google Search, then JAWS will announce that Google search. So it would be easy for a blind person to recognize things in an easy manner.

[](https://www.guru99.com/images/cassandra/021316_1225_Whatisacces5.png)

The point I want to explain here a screen reader will narrate word by word if you enter something or in the text box**.**Similarly, if there is link it will pronounce it as a link, for Button it will pronounce it as a button. So that a Blind person can easily Identify things.

Now If a website is poorly designed and developed, then it might be possible (it generally happens) that jaws would not be able to narrate correct content which in turn result for inaccessibility for Blind Person.(Say if jaws are narrating a link as a content, then a blind user would never able to know that it's a link and if that would be a crucial one for that website then ????).In that case, it would be a result into a high loss for Website Business.

**2) Visual Impairment**

There are two categories which I want to be mentioned under visual impairment.

The first one is Color Blindness. Color Blindness means not completely blind but not able to view some specific color properly. Red and Blue are the common colors which people not able to see properly if they do have color blindness. So basically, if I do have a color blindness of red color and I want to use website which is 80% in red then???Would I be comfortable on that website? The answer is No.

So a website should be designed such that a person with color blindness does not have any problem to access that. Take a simple example of a button which is in Red. To make it accessible if it is outlined with Black. Then it is easy to access. Normally Black and white are considered as universal.

**3) POOR VISION DISABILITY**

The second thing is a person having poor vision (not clear vision) or having different eyesight problem (there are many eye problem related to the retina, etc.) for accessing any site.

1) In such cases, the best thing to do is avoid small text. Because it would be a great advantage for poorly vision people.

2) Also, people with vision problem would like to zoom text of website to make it comfortable for them. So a website should be designed in such a manner that if enlarging it, its layout is not breakable when zooming the text. Otherwise, it won't be a good impression for them.

**4) Other Disability**

In Accessibility Testing for Disabled audience one very major point to consider is Accessing the Website without the use of the mouse. A person should be able to complete access the website the links, buttons, radio buttons, checkboxes, pop-ups, dropdown, all the controls should be completely accessible and operable through the keyboard.

**For Example**: If I am right handed paralyzed, and I am not comfortable with a mouse or say I don't want to use a mouse then what? In that case, if I am not able to access link or checkboxes on site via keyboard then???? So a website should be completely accessible with Keyboard.

Alternative Text should be there for Images, Audio, Video so that screen reader reads them and will narrate them so that a blind person can easily recognize what the image, audio, the video is all about. In addition, to it, keyboard shortcuts should be there to easily access website and navigation should be available with the keyboard.

Also, the focus should be completely visible. When we are pressing tab, then the user should be able to see where the control is moving. With visible focus, it becomes very easy for a user having poor vision or color blindness to identify the flow of a site and also an ease of access.

**User with Hearing Disability (Deaf or hard to listen):**The last ones are a person having a disability of Hearing. A deaf person can access the website as he is what able to see the content on the website. But when it comes to audio and video they face difficulties. So in that case, for any Video and Audio, there should be Alt text. Alt text means Alternative text. Suppose there is any Video about how to Book an airline ticket. In that case, the text should be there so that a deaf person can read that and get the idea what the video is all about.

**Accessibility Testing Tools:**

To make your website more acceptable and user-friendly, it is crucial that it is easily accessible. There are various tools which can check the accessibility of the website. Some of these popular tools are listed below-

**1) Wave**

[](https://www.guru99.com/images/5-2015/050115_1151_Top5WebAcce2.jpg)

Wave is a free web accessibility tool created by WEBAIM. It is used to validate the web page manually for various aspects of accessibility. This tool can be used to check the intranet, pass-word protected, dynamically generated, or sensitive web pages. Major functions of Web Accessibility Toolbar includes identifying components of a webpage, providing access to alternative view of page content and facilitating the use of third party online applications. It ensures 100% private and secure accessibility reporting

Visit [here](http://wave.webaim.org/)

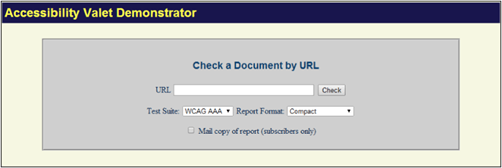
**2) TAW**

[](https://www.guru99.com/images/5-2015/050115_1151_Top5WebAcce3.jpg)

TAW is the online tool for determining accessibility of your web. This tool analyzes the web site in accordance with W3C web accessibility guidelines and shows accessibilities issues. Web accessibility test issues are categorized into priority 1, priority 2 and priority 3. The interesting feature of TAW is the ability to generate subsets of WCAG 1.0 to test against. In TAW tool, you can either choose to test a single page or multiple pages by "spider" a site. TAW also enable us to define additional checks via the "User Checking's" dialog box

Visit [here](https://www.tawdis.net/)

**3) Accessibility Valet**

[](https://www.guru99.com/images/5-2015/050115_1151_Top5WebAcce4.png)

It is a tool that allows you to check web pages against WCAG (Web Content Accessibility Guidelines) compliance. All the HTML reporting options show your markup in a normalized form highlighting deprecated, bogus and valid mark as well as elements that are misplaced. This tool offers various features like

* In-dept reports for developers
* Executive summary for QA and Management
* Meta-data for the semantic web and WWW
* Automatic cleanup and Html to XHtml conversion
* Scripting tools

Visit [Here](http://valet.webthing.com/access/url.html)

**4) Accessibility Developer Tools**

It is a Chrome extension. It does and accessibility audit. The results of the audit show accessibility rules that are violated by the Page Under Test. The extension has high reviews and is frequently updated

Visit [Here](https://chrome.google.com/webstore/detail/accessibility-developer-t/fpkknkljclfencbdbgkenhalefipecmb?hl=en)

**5) Quick Accessibility Page Tester**

Since there are some excellent accessibility toolbars, Quick Page Accessibility Tester is a bookmark that you can click to get a quick analysis of the web page. It will figure out various issues with your page, warns about possible issues and highlight areas on the page which might benefit from ARIA (Accessible Rich Internet Applications).

Visit [Here](http://accessify.com/tools-and-wizards/accessibility-tools/favelets/quick-page-test/)

There are various tools available in the market to perform web accessibility testing given below:

**6) aDesigner**

This is a tool developed by IBM which simulates the experience of visually impaired individuals so that the designer can better understand the needs of disabled people and develop applications accordingly.

Visit [Here](http://www.eclipse.org/actf/downloads/tools/aDesigner/)

**7) WebAnywhere**

This is a browser based tool that works similarly to the screen readers like Jaws. It assists the readers how to read the web page.

Visit [Here](http://webanywhere.cs.washington.edu/)

**8) Web accessibility toolbar**

WAT is an extension of Internet explorer or Opera that offers web page designers with useful features in the analysis of web page. One best feature is GreyScale feature which helps to find low contrast spots in the design.

Visit [Here](https://developer.paciellogroup.com/resources/wat/)

**Myths of Accessibility Testing:**

Following are the Myths of Accessibility Testing:

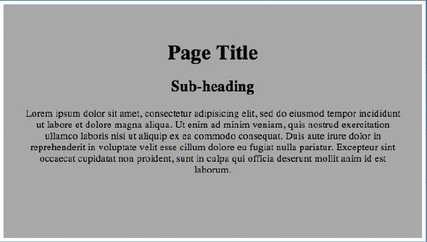
**Myth:**Creating Accessible website is expensive

**Fact:**It is not expensive. Take the timeout to think about accessibility issues in the design stage itself along with basic testing .This will save money as well as rework.

**Myth:**Changing inaccessible websites to accessible website is time consuming and expensive

It is not necessary to incorporate all the changes at one time. Work on basic needs which are most necessary for disabled users.

**Myth:** Accessibility is plain and boring

[](https://www.guru99.com/images/AccessibilityTestingIsBoring.png)

**Accessibility doesn't mean text only page**

You can make web pages attractive, but it should be designed in such a way that it can be accessible by all users. Also as per W3C web content accessibility guidelines - it strongly discourage the use of text only pages.

**Myth:**Accessibility if for the Blind and Disabled

**Fact** Following accessibility guidelines improves the overall usability of the software, which helps regular users as well.

**Conclusion**

In Software Engineering, Accessibility testing helps in making your application disabled friendly. If following accessibility guidelines is not possible due to complexity of your web application, build one version of the website for regular users and other for disable