Chapter-3

THEOROTICAL FRAMEWORK OF TESTING

3.1 Theoretical Concepts of testing

3.1.1 Software Testing

Some of the Traditional Definitions of Software Testing are

1. Process of demonstrating that errors are not present.
2. It is measurement of Software Quality.
3. Testing is process to prove that the Software works correctly.
4. Evaluating program or system.

3.1.2 Complete Definition of Software Testing

"It is process of executing a program to find out an error as early as possible during SDLC." A successful test must find at least one error. Testing is process not single activity.

3.1.3 ISTQB Definition for Software Testing

The process consisting of all life cycle activities both static and dynamic concern with planning, preparation and evaluation of software products and related work products to determine that they satisfy specified requirements, to demonstrate that they are fit for purpose and to detect defects.

Here, planning means designing test plan. Preparation means designing test cases

3.1.4 Testing Objective

Testing activity should start as early as possible in SDLC and should be focus on different objective

1. Finding Defects
2. Preventing Defects
3. Ensure the product is bug free before release.
4. To demonstrate given software product matching its requirement

3.1.5 STLC:Software Test Life Cycle

STLC have following 8 phases.

- Establish test Objective
- Prepare test plan
- Design test Cases
- Review Test Cases
- Execute tests
- Prepare test report
- Examine test reports
- Perform postmortem Reviews

3.1.6 Testing Principals

1. A programmer should avoid testing his or her own code.
2. A team should avoid testing its own program
3. Thoroughly inspect result of each test

Good Testing must examine: User Requirement and Design objectives.

3.1.7 Test levels

3.1.7.1. Unit Testing

It is done by Developer at unit level. In unit testing individual components are tested to ensure that they operate correctly. The idea behind unit testing is to write test case for every function and method in the module. Automation tools used for Unit Testing. Turn: For C, C++ Projects, Xunit.Net for .Net Projects.

Advantages of Unit Testing:

1. To check either extreme programming is done or not.
2. To ensure each and every unit of program is working correctly.

Limitations of Unit Testing:

1. It only test the functionality of unit themselves
2. It will not catch integration errors, performance problems and any other system wide issues.

3.1.7.2. Module Testing:

It is done on related unit testing components.It is collection of dependent components of procedures and functions. A module is independent entity in software. The tested units are integrated into module and each module is tested separately for

specifications. If the modules are independent these modules can be tested parallel. Testing multiple modules or sub modules in parallel is called as parallel testing.

3.1.7.3. Integration testing:

It starts at module level. Integrate all individually tested modules and test them together to form a sub system. In this testing more stress is given on interfaces between the modules.

Two approaches of integration testing

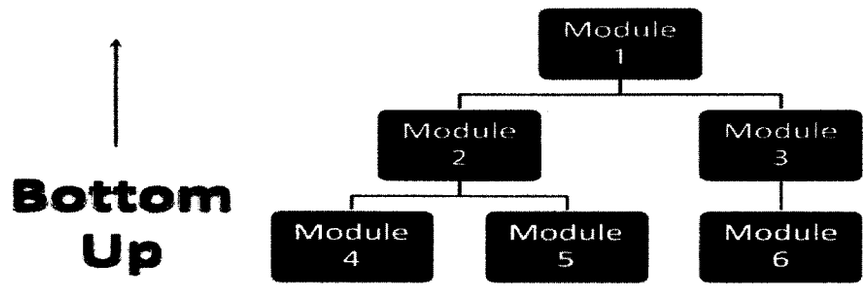
1. Bottom up 2.Top down

1. Bottom up approach:

The program is combined and test bottom of tree to top. Each component at lowest level of system hierarchy is tested individually first. Then next components to be tested. In bottom up testing you write your own modules called test drivers that exercise the module you are testing. They look in exactly the same way that the future real module will, then test driver send the test case data to module under test, read back the result and verify that they are correct. You can thoroughly test the software using this way feeding it all types of data.

Test Driver: A routine that calls the particular Component and passes test case to it.

Diagrammatic Representation:



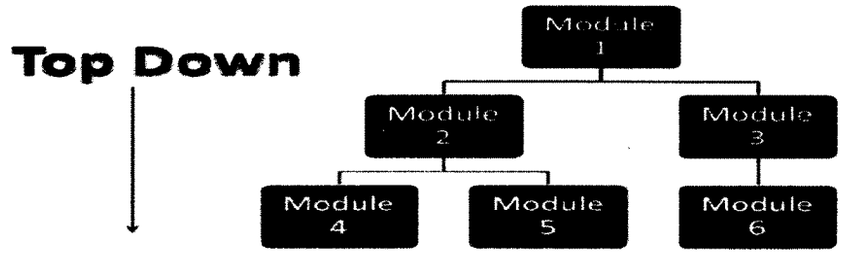
Limitations of bottom up approach:

1. Top level components are usually most important but last to be tested.
2. It is most sensible for object oriented programming.

2. Top down approach:

The top level usually one controlling Component is tested first by itself then all the components called by tested components are combined and tested as larger unit. A test stub sends the test data up to module being tested.

Diagrammatic Representation:



Limitations of top down approach:

1. Writing stubs can be difficult because they must allow all possible conditions to be tested.
2. Stubs may itself needed to be tested to ensure it is correct.

3.1.7.4. System Testing

It starts at integration level. It validates that the system meets its functional and non-functional requirements. In addition to functional testing the system has to be tested for non-functional requirements such as-1.Performance requirement. 2. Security 3.Safty 4.Portability 5.Interoperability with hardware and software.

3.1.7.5. Acceptance Testing

It is final stage of testing before the system is accepted for operational use. It is done with the data supplied by client. After the system testing is completed the software has to be approved by the customer. The customer will carry out rigorous testing called as acceptance testing and if there are no errors found the customer will pay the money. It is done either at developers or client side.

According to this it is classified in two ways:

3.1.7.5.1. Alpha testing-

The software is tested at developer's site by customer. The users and members of developers Organization are invited to use the system. Developer observes the user and notes the Problems. It is also carried out by independent testing team. Since the developer of organization is present so all the tests conducted in Controlled environment.

3.1.7.5.2. Beta testing:

Beta testing is conducted at one or more customer site by end user of software. Give the system to end user who install it and use it in the real world working condition. He records the bug from the system and sends this report to developer of organization. Since the developer is not present the live application environment is not controlled by Developer.

3.1.8 Test Types

3.1.8.1. Black box Testing

It is also called as specification based, functional, behavioral, closed box, opaque testing. It is totally based on or focused on testing for requirements and functionality of software application.

Black Box testing is procedure to derive or select test cases based on analysis of specification either functional or non-functional of component or system without reference to its internal logic.

3.1.8.2. White box Testing

Designing Test Cases based on internal structure of program is called White box testing.

3.1.8.3. Regression Testing

Testing is done to determine that weather the changed components have introduced any error in unchanged component. Whenever change is made in software run previously designed test cases again to ensure that unchanged components function

correctly. Also tester has to review previously prepared documents to ensure that they remain correct.

Advantages of Regression Testing

1. All the aspects of system remain functional after testing
2. Change in one segment does not change the functionality of other segment.

Disadvantage:

Time consuming and tedious if test automation is not done.

3.1.8.4. Retesting

Retesting is done to ensure that the fault is rectified by running the same test case repeatedly.

Example: Tester founds the bug. Tester will assign that bug to developer. Developer will fix that bug. Again tester will retest that bug by executing same test case. but in retesting tester will not check due to that change any other software part is affected.

3.1.8.5. Smoke Testing

Testing the software without any user input is called as smoke testing.

It involves basic functionality check.

For example: Installation, Navigation through the application, invoking or accessing all major features. This is also known as BVT (Build verification testing).

3.1.8.6. Sanity Testing

Testing the software with some user input. It will test the behaviour of application.

Example: To test the connectivity to the database, application servers, printers etc.

3.1.8.7. Usability Testing

To test either software developed is user friendly is called usability testing. The main focus of usability testing is to test UI.

User Interface Testing:

This means that you used to interact with software program is called its user interface. All softwares have UI. The important points to test UI

1. Follows Standards and Procedures.

2. Flexible

3. Correct.

4. Consistant

5. Comfortable

1. Follows Standards and Procedure.

Your Software must follow the standards and procedure for particular platform.

Example: Ms-windows displaying three types of Message box. Information Message (i), Warning message (!) and critical message(X).When and how to use each one is defined in UI. Standards for windows. These standards and guidelines are developed by experts in software usability.

2. Flexible:

User like choices not too many but enough to allow them to select what they want to do and how they want to do it.

For Example: Windows calculator has two views. Standard and scientific. Users can decide which they need for their task. The windows calculator showing its flexibility by having two different views.

3. Correct

Testing the correctness means that test weather UI is doing what it is supposed to do. So for correctness in UI tester has to test following things.

1. Language and spelling

2. Bad Media: Media means any supporting icon, images, sounds, videos that go with your software UI. Icon should be of same size, sound should all be of same format.

3. WYSIWYG (What you see is what you get)

Means when you click on save button is the document on the screen exactly saved to the disk?

4. Useful: Test either UI is useful for users. Either it is easy to understand.

4. Consistant

Users develop habit and expect that if they do something in certain way in one program another will do same operation in same way.

For consistency test-

1. Terminology and naming:

Are same terms used throughout software? Are features named consistently?

2. Placement and Keyboard equivalents for buttons.

5. Comfortable

Software should be comfortable to use. It shouldn't make it difficult for the user to do his work. It can be difficult to find out comfort but following few things will give you idea of how to identify good or bad software comfort..

1. Appropriateness:

Software should look and feel proper for what its doing.

Eg: Financial business application should not designed with more colours and sound effects.

2. Error Handling:

Program should handle appropriate Errors. Also program allows the user to restore data lost because of mistake.

3. *Performance:*

If operation is slow it should at least give the user feedback on how much longer it will take and shows that it is still working.

3.1.8.8. Installation Testing

Testing the installation (setup program) on different operating system, hardware Configurations before releasing the software are called as installation testing.

Installation testing tips with some broad test Cases-

1)Use flow diagrams to perform installation testing. Flow diagrams simplify our task.

2) If you have previously installed compact basic version of application then in next test case install the full application version on the same path as used for compact version.

3) If you are using flow diagram to test different files to be written on disk while installation then use the same flow diagram in reverse order to test uninstallation of all the installed files on disk.

4) Use distributed testing environment in order to carry out installation testing. Distributed environment simply save your time and you can effectively manage all the different test cases from a single machine.

5) Test disk space requirement on different file system format. Like FAT16 will require more space than efficient NTFS or FAT32 file systems.

6) Test the installer scripts used for checking the required disk space. If installer is prompting required disk space 1MB, then make sure exactly 1MB is used or whether more disk space utilized during installation. If yes flag this as error.

7) Use software's available freely in market to verify registry changes on successful installation. Verify the registry changes with your expected change list after installation.

8) Forcefully break the installation process in between. See the behavior of system and whether system recovers to its original state without any issues. You can test this "break of installation" on every installation step.

9) Disk space checking: In manual methods you can check free disk space available on drive before installation and disk space reported by installer script to check whether installer is calculating and reporting disk space accurately. Check the disk space after the installation to verify accurate usage of installation disk space.

10) As you check installation you can test for uninstallation also. Before each new iteration of installation make sure that all the files written to disk are removed after uninstallation. Some times uninstallation routine removes files from only last upgraded installation keeping the old version files untouched. Also check for rebooting option after Uninstallation manually and forcefully not to reboot.

11. Installation over the network

12. Online installation

13. Database checking on Installation,

14. Shared DLL installation and uninstallation etc.

3.1.8.9. Performance Testing

In software engineering, **performance testing** is testing that is performed, from one perspective, to determine how fast some aspect of a system performs under a particular workload.

Performance testing can serve different purposes.

1. It can demonstrate that the system meets performance criteria

2. It can compare two systems to find which performs better.

3. It can measure what parts of the system or workload cause the system to perform badly.

Purposes

- Demonstrate that the system meets performance criteria.
- Compare two systems to find which performs better.
- Measure what parts of the system or workload cause the system to perform badly.

Setting performance goals:

Performance goals will differ depending on the application technology and purpose however they should always include of the following:-

1. Concurrency / Throughput:

If an application identifies end-users by some form of login procedure then a concurrency goal is highly desirable. By definition this is the largest number of concurrent application users that the application is expected to support at any given moment.

2. Server response time:

This refers to the time taken for one application node to respond to the request of another. A simple example would be a HTTP 'GET' request from browser client to web server. In terms of response time this is what all load testing tools actually measure.

Performance Testing Web Applications Methodology

According to the Microsoft Developer Network the Performance Testing Methodology consists of the following activities:

Activity 1: Identify the Test Environment.

Identify the physical test environment and the production environment as well as the tools and resources available to the test team. The physical environment includes hardware, software, and network configurations. Having a thorough understanding of the entire test environment at the outset enables more efficient test design and planning and helps you identify testing challenges early in the project.

Activity 2: Identify Performance Acceptance Criteria. Identify the response time, and resource utilization goals and constraints. Additionally, identify project success criteria that may not be captured by those goals and constraints; for example, using performance tests to evaluate what combination of configuration settings will result in the most desirable performance characteristics.

Activity 3: Plan and Design Tests. Identify key scenarios; determine variability among representative users and how to simulate that variability.

Activity 4: Configure the Test Environment. Prepare the test environment, tools, and resources necessary to execute each strategy as features and components become available for test.

Activity 5: Implement the Test Design. Develop the performance tests in accordance with the test design.

Activity 6: Execute the Test. Run tests. Validate the tests, test data, and results collection.

Activity 7: Analyze Results, Tune, and Retest. Analyse, Consolidate and share results data. Make a tuning change and retest. Improvement or degradation? Each improvement made will return smaller improvement than the previous improvement.

3.1.8.9.1. Load Testing

A test type concern with measuring the behaviour of application or component or system with increasing load.Load testing is the process of putting demand on a system or device and measuring its response.

For example: If a web site with a shopping cart is intended for 100 concurrent users who are doing the following functions:

1.25 Users are browsing through the items and logging off.

2.25 Users are adding items to the shopping cart and checking out and logging off.

3.25 Users are returning items previously purchased and logging off.

4.25 Users are just logged in without any activity.

So you have to create 100 virtual users (25 on each system) using automation tools and observe the performance.

The pass fail criterion is different for each individual organization and there are no standards on what an acceptable criterion should be, across the board. It is a common misconception that these are record and playback tools like regression testing tools. Load testing tools work at the protocol level whereas most regression testing tools work at the GUI object level

Popular load testing tools:

Tool Name	Company Name
OpenSTA	Open System Testing
The Grinder	Open-source tool.
IBM Rational	IBM Performance Tester
JMeter	An Apache Jakarta open source project.
LoadRunner	HP
SilkPerformer	Micro Focus

3.1.8.9.2. Stress Testing

Stress testing is a form of testing that is used to determine the stability of a given system or entity. It involves testing beyond normal operational capacity, often to a breaking point, in order to observe the results.

3.1.8.10. Recovery Testing

Recovery testing is the activity of testing how well an application is able to recover from crashes, hardware failures and other similar Problems.

Examples of recovery testing:

1. Restart the system while a browser has a definite number of sessions. Afterwards, check that the browser is able to recover all of them.
2. While an application is running, suddenly restart the computer, and afterwards check the validness of the application's data integrity.
3. While an application is receiving data from a network, unplug the connecting cable. After some time, plug the cable back in and analyze the application's ability to continue receiving data from the point at which the network connection disappeared.

3.1.8.11. Documentation Testing

Testing the documents throughout SDLC is called as Documentation Testing.

If the documentation is not right: there will be major and costly problems.

a. Start Documentation testing at the very beginning of the software process and hence save large amounts of money, since the earlier a defect is found the less it will cost to be fixed.
b. Ensure that documents are formally reviewed, Tested.

c. Determine the completeness of individual Documents.
d. Test Documentation content.

3.1.8.12. Monkey Testing.

Monkey /Gorilla test technique aim is to break the application under test. A test technique commonly used in Unit testing and system testing phase in which the tester acts as a monkey (not in the literal sense) by giving random input across each and every unit/component or module to check the stability and workability of the system.

a. It is a error-based test technique that involves extensive testing of system's functionality with random valid or invalid inputs.
b. Many organizations name Monkey/Gorilla Testing as reliability or fault tolerance testing

3.1.9 Testing of web applications & Quality assurance of web applications.

A Web application is an application that is invoked with a client (mostly by Web browser).over the Internet, Intranet or Extranet. A Web-based application allows the information processing functions to be initiated remotely from a client (browser) and executed partly on a Web server, application server and/or database server. These applications are specifically designed to be executed in a Web-based environment. When we visit on Web, we can find different kind of Websites. In general, there are two types of Websites, the one type is based on the HTML also called static Websites and behave like simple printed newspapers or magazines. These Websites have published and printed materials for the end users. The examples of such kind of Websites are the different Websites of newspapers e.g. The New York times, BBC etc. The second type of Websites enables the end users to interact with the Website. In this type Web pages are generated dynamically in the response of end user's input or action. These Websites work as software and utilities, also called as Web applications. Web applications (also called Web software) run on servers and end users access these applications through Web browsers. The examples of Web applications are supply chain management, online banking systems, online retail systems and different email services like Google, yahoo and hotmail. Web applications are more complicated as compare to simple static Websites and provide a new way to deploy software applications to the end users. Web-based applications are based on mixture

between print publishing and software development, between marketing and computing, between internal communications and external relations, and between art and technology. Web applications are interactive software which has complex Graphical User Interfaces (GUIs) and numbers of back-end software components are integrated. These applications have revolutionized the business arena and have provided new opportunities to businesses and to the end users. Web applications utilize different novel technologies, tools, programming models, and programming languages to fulfill the high quality requirements. These applications also utilize servers, browsers, and usually internet to reach the end users. The better way to define Web application is in terms of its components. The components that include in Web application are:

- **Web Server:** the computer that delivers Web pages
- **Application Server:** a server and a program that handles all the operations of backend computing applications (like databases) and end users.
- **Front-end Systems:** the user interface for end users
- **Back-end Systems:** different applications and databases used by the end user to access or update program.
- **Code, Business Logic, Static files:** programs and parameters that instruct servers and systems on how to process information

3.1.9.1 Characteristics of Web-based Applications

Every software application has their specific characteristics, some of those are common and some are related to specific applications. These characteristics are very important to keep in mind before the development of any application. Like other software applications, Web based applications have characteristics which act as important factors in these applications. There are some general characteristics that apply to all Web applications but with different degrees of influence. The degrees of influence vary with different categories of applications. The following general characteristics and attributes are encountered in majority of Web-based applications:

A] Network intensive

Since Web applications are delivered to a diverse community of users, the nature of Web based Applications is network intensive. These applications reside on network (like Internet, Intranet or Extranet).

B] Content Driven

Mostly Web applications present textual, graphical data, audio and video to the end users by using hypermedia.

C] Continuous Evolution

Most Web applications evolve continuously. These applications updated on the regular interval, even some applications are updated on hourly schedule to provide latest information to the end users.

D] Short development schedule

Mostly Web-based applications have very tight development schedule. It means these applications have very short time for the development and developed under compressed time schedule. The time to market for a complete Website from planning to implementation and testing can be a matter of few days or weeks.

E] Security

To protect sensitive content and information provided by the user, and for successful data transmission strong security measures are implemented in the Web-based applications. One of the most important characteristics of web applications is aesthetic appearance. Aesthetic appearance of those Web-based applications which designed for selling products and ideas is as important as technical design. The above all are some simple but important characteristics of Web-based applications, but as the complexities of these applications are growing, some other characteristics such as distributed, heterogeneity, autonomous, dynamic, hypermedia, multiplatform support, ubiquitous are very important to understand. These characteristics are the important factors that are necessary to keep in mind before the designing, implementing, testing and deployment of the Web-based applications. These characteristics can assist the developers and software engineers to built successful applications.

3.1.9.2 Types/Categories of Web-based Applications

Along with characteristics of Web-based applications the understanding of different types and categories of these applications is also important for the developer to develop successful applications. The Web-based applications are of many types each of which has its own issues and importance. The following different categories of Web applications are mostly present on the Web.

• **Interactive:** These types of applications are usually provides the mutual interaction and communication way among the community of users like chat rooms, instant messaging etc.

• **Informational:** In these types of applications read-only information is provided to the end users through different and simple navigations and links.

• **Customizable:** Through these applications end users can customize the contents of the applications according to their needs and preferences. For instance, a user can customize email settings according to its needs and preferences.

• **Download/ Deliverable:** These kind of applications provide the facilities of downloading different applications, files etc. For example, if a user wants to upgrade or update Microsoft Windows, he/she can do it through the Web applications provided by the Microsoft.

• **Form-based Input:** Through these applications end users can submit their data, queries to the database of the organization and extract the required information.

• **Transactional:** Through these Web applications users make the request or place the order to obtain goods or services. For instance online shopping, online ticketing purchase.

• **Web Services:** Web services allow us to create the interoperable distributed applications. These are cross-platform technology and can be developed in multiple technologies and make the data available to different applications that run on different platforms. Familiar Web Services include Business to Consumer (B2C), Business to Business (B2B), search engines, stock tickers, FedEx tracking, and credit card services etc.

• **Web Portals:** These kinds of applications provide the facilities to the users to other contents of the Web or services which are not in the domain of the portal applications.

• **Data Warehouses:** These applications provide the facilities to the user to query their request in the collection of large databases to retrieve and extract particular information. Google is a good example of such kind of applications.

3.1.9.3 Quality Assurance of Web-based applications

The success of software engineering depends upon the delivery of high quality software. Quality is one of the key factors in the market growth and success of a product. In recent years, quality of software product and quality in service has become

principles for many corporations and organizations to distinguish themselves from competitors and to cover larger market place. Quality is an ambiguous word; means there are lot of definitions available for quality. According to IEEE, quality is the degree to which software meets customer or user needs or expectations.

The simplest definition of quality is in the mean of customer satisfaction. Robert Glass summarize the customer satisfaction in mathematical equation as

Customer satisfaction = compliant product + good quality + delivery within schedule and budget

He also argues that quality is an important factor in the development of a product, but if the customer is not satisfied then nothing else really matters.

Quality assurance includes all the process-related activities to achieve the quality. It is involved from the start of a project. In other words we can say that it is an umbrella activity which is applied on each step in the software development process. It controls the insight and outsight quality of the software. Software quality assurance includes the following important elements and methods.

- Quality management approach
- Effective software engineering methods and tools
- Formal technical reviews applicable to whole software process
- Effective testing strategies and techniques
- Procedures to control documentations and changes in it
- Procedures to assure compliance to standards
- Mechanisms for measurement and reporting

In 1990's mostly Websites were typically static and composed only of Web pages stored in some file systems that were connected together through hyperlinks. The main aim of Websites was simply to provide information across the Web in a simple plain and intuitive way. Therefore at that time, quality assurance was a relatively unchallenging and simple task. Now Web has become the essential part to many software applications in various parts of the businesses and organizations. The dependency of people on Web applications is increasing continuously due to which the Web systems have become more and more complex. These applications are increasingly integrated in business strategies of small and large organizations. Therefore quality, reliability, accessibility, usability, adaptability and functionality

have become very crucial and important factors for the Web applications. The process of Web engineering is used to develop the high quality Web applications. It defines the specific techniques, methodologies and models to develop Web-based applications. The main aim of engineering of Web-based applications is to attain and produce high quality software products. Quality assurance of Web-based applications is very crucial and vital to achieve the high quality. The quality assurance of Web applications is the responsibility of Web developers and Software quality assurance group. To ensure the quality of Web applications the development team should follow the above mentioned methods of software quality assurance. That is, Web developers should follow the quality management approach, effective software methods and tools, formal reviews, effective testing strategies and techniques, follow the standards and usage of appropriate mechanisms for measurement and reporting.

3.1.9.4 Quality attributes of Web-based applications

Different persons (end-users of Web applications) have different views and opinions about the good Web Application. These opinions and views depend upon the end user and vary widely, because some individuals like flashy graphics and some want simple text. Some users want detailed information and some only like short and abbreviated presentations. It is fact that the user perception of likeness of Web application might be more important that any technical discussion of Web applications quality. This raise the question about the perception of quality of Web application and about the different attributes that must be exhibit to achieve goodness in the eyes of the end-users and also exhibit the technical characteristics of quality that enable the Web engineers to enhance, adapt, correct and support the Web application over the long term. Almost all general quality characteristics can be applied to Web applications but the most important and relevant quality attributes are prepared, who developed a quality requirement tree that identifies a set of attributes that lead to develop high quality Web applications. The figure 8 shows this quality requirement tree for high quality Web applications.

3.1.9.5 Quality assurance enabling technologies

In order to build reliable and high quality Web applications the Web engineer should be familiar with the quality assurance enabling technologies. These enabling

technologies are component based development, internet standards and security.The brief description of these enabling technologies is as under:

A] Component based Development

The explosive growth of Web-based applications has evolved the component technologies. The available famous infrastructure standards for web development are CORBA (Common Object Request Broker Architecture), COM/DCOM (Component Object Model/ Distributed Component Object Model) and JavaBeans. These different standards are helpful to deploy and integrate third party components and to develop custom components to communicate with each other and with other system services.

B] Internet Standards

Internet standards are specifications which are stable and well-understood, has multiple, independent, and interoperable implementations with operational experience and recognizably useful in some or all elements of the Internet [30]. In early 1990's HTML (Hyper Text Markup Language) was the dominant standard to develop Web applications but now the Web applications have become more complex, their size is growing, therefore new standards have emerged. XML (Extensible Markup Language) and XHTML (Extensible Hypertext Markup Language) are new standards that have been adopted to develop next generation Web applications. These standards allow developers to define their own custom tags to describe the content of Web pages in Web applications. By following these standards the quality of Web applications is increased in the mean of robustness, interoperability, integration, functionality, reliability and accessibility.

C] Security

When our application deployed and launched on the network or Internet, there are great risks of unauthorized used. There are great threats of vulnerabilities. The hackers try to unauthorized access in the intent of some profit or for some other aims. Sometimes internal personnel can be involved in unauthorized access of particular application for their specific benefits and aims or malicious intents. Therefore security measures are very important to build high quality Web applications. A lot of security measures are being applied to minimize the threats of vulnerabilities and malicious use of the particular applications like firewall, encryption, and other security policies.

3.1.9.6 Testing of Web-based applications

Along with other quality assurance activities, one of important and critical element or component of quality assurance is testing. Testing has great importance to develop high quality software products and its importance and implications cannot be overemphasized. Testing represents the final review of specification of the system, design of the system and implementation of the system. According to [1], Testing is the process to execute and inspect the program with the intent of finding errors. Web applications have become the crucial components of our lives. The importance of these applications cannot be overemphasized. These are crucial vehicles for information exchange, commerce, and a host of social and educational activities. The rapid evolve, complex and ubiquitous nature of Web based applications are the major hurdles to achieve the high 22 quality. Mostly ad hoc methods are used to achieve the quality of Web based applications. These methods and techniques are important to understand and improve the quality but due to the growing complex nature of Web-based applications more systematic methods and tools are required. Different formal technical reviews and other important activities of quality assurance are helpful to uncover the errors but are insufficient. Testing is considered as tedious procedure in the Web applications development and mostly it is neglected by the Web engineers due to the short development schedule of these applications. But it is fact that today most organizations rely on Web applications more than ever to provide manage information with strategic and operational importance, and services to customers and end-users. This increase the awareness of the importance and of testing Web applications as a critical and crucial activity. The testing is the key process and activity of the quality assurance and in the development of Web-based applications. The process of systematic and well planned testing has the greater possibility to uncover the errors and to ensure the high quality of Web-based applications. Testing is the most applicable method to verify the performance and functional requirements of the applications. It is one of the most important processes (phase) in the software development life cycle because it ensures the quality, stability, adoptability, sustainability and acceptability of Web-based applications. In other words we can say that it include all verification and validation activities of the Web applications.

3.1.9.7 Testing Challenges to Web-based applications

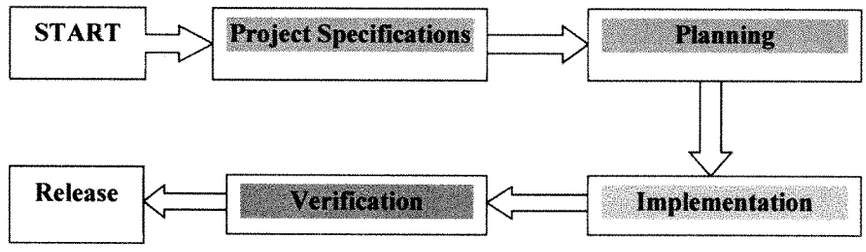
The main aim of testing Web-based applications is not different from the testing of conventional software applications. That is to find out the errors and bugs in the applications before deployment on Internet, or Intranet. Due to the enormous involvement of different interdependence Web components, the complex nature of applications, heterogeneity and distributed nature, we can find plenty of bugs and errors in Web applications. We can find more errors but how? It's a very big challenge in Web-based applications because the importance of rooting out errors in Web applications cannot be understood. Due to the large number of different elements and a lot of interdependent components of Web-based application the Web engineer faces many challenges during the designing and testing of these applications. Many assumptions about the environment, platform and other resources are required. Web-based applications are error prone and have many failure points that must be considered before designing and deploying the testing approach.

3.2 Technical Contribution of researcher

3.2.1 Testing framework

As per the researcher opinion, the follwing testing framework will be useful for testing team to test the project effectivly and more systamatic way.

Proposed Testing Framework



The framewrok is basically devided into four phases

1. Project Speciifcation
2. Planning
3. Implementation
4. verification..

The details of activities need to be planned during these phases are as follows:

1.Project Specification:

During project specification phase team leads or test managers should focus on follwing activities.

a. Gather requirements

b. Create test plans from requirements

c. Define test approach.

Use possible testing techniques as per the project requirements and define test matrics

2.Planning :

During planing phase team leads or test managers should focus on follwing activities.

a. Break out requirements into tasks.

b. Break out the requirements into scenarios,use cases,quality of service,Test cases

Identify test cases flow detail

c. identify test environment

3.Implementation:

During implemenation software tester write different types of test cases

a. Write functional test

Understand function under test and write test cases

b. Write security test

Explore entry points and identify system weakneses

c. Write performance test

Understand objective of performance testing accordingly design test and document test.

d. Write Load test

Understand the objective of load test and design and document load test.

e. Write stress test

Design and write stress test

4.Verification:

The test designed during implementation phase of framework need to be excuted in verification phase

For that deploy the application and run the test and analyse the results,if result is pass then close the work otherwise enter the defect in any open source defect tracking tool and assign severity and priority to defect.Finally after fixing of all defects and completion of all planed testing activities according to test plan then relese the product.

3.2.2 Critical factors need to consider while Usability testing of web Application

To test the UI of website is called as Usability testing. To test either your UI is user friendly for user to do their tasks. The following **factors needs to consider while testing UI of** website as **per the researcher recommendation**

1. Navigation

Navigation describes the way users navigate within a page, between different user interface controls (buttons, boxes, lists, windows etc.), or between pages via e.g. links.

a. Application navigation is proper through tab & mouse
b. Main features accessible from the main/home page.
c. Any hot keys, control keys to access menus.
d. Navigation Labels Are Clear & Concise
e. Company Logo Is Linked to Home-page-This may sound minor, but people expect logos to link to home-pages, and when they don't, confusion follows. Researcher seen video of users clicking on a logo over and over, with no idea what to do next.
f. Links Are Consistent & Easy to Identify
g. Site Search Is Easy to Access-If you have a site search, make sure it's prominent. Usability guidelines tend to prefer the upper-right corner of the page.

2. Content

Correctness is whether the information is truthful or contains misinformation. The accuracy of the information is whether it is without grammatical or spelling errors. Remove irrelevant information from your site. This may otherwise cause misunderstandings or confusion. The points to be tested,

a. Spellings and Grammars
b. Updated information's
c. General Appearance

d. Page appearance

e. Color, font and size

f. Frames

g. Consistent design

h. Major Headings Are Clear & Descriptive-Headings should be clear, and for SEO benefit, using heading tags

i. URLs Are Meaningful & User-friendly-This is a point of some debate, but meaningful Keyword-based URLs are generally good for both visitors and search engines.

j. HTML Page Titles Are Explanatory-More importantly, your page titles (in the <TITLE> tag) should be descriptive, unique, and not jammed full of keywords. Page titles are the first thing search-engine visitors see, and if those titles don't make sense or look spammy, they'll move on to the next result.

3. Identity-A key question when someone first comes to your site is "Who are you?" It's important to answer it quickly. The point to be tested in identity

a. Clear Path to Contact Information-Web Application Visitors want to know that they can get in touch with you if they need to. It's also hard to do business if no one can contact you. Preferably, list your contact information as text (not in an image) - it'll get picked up by search engines, including local searches.

b. Company Logo Is Prominently Placed-Put your logo or brand where it's easy to find, and that usually means the upper-left of the screen. People expect it, and they like it when you make their lives easy.

c. Home-page Is Digestible In 5 Seconds-website visitors are a fickle bunch, and they need to get the basic gist of your home-page in just a few moments.

d. Tagline Makes Company's Purpose Clear

4. Accessibility-It means anything that might keep a visitor from being able to access the information on a website. The points to be tested-

a. Web Application Load-time Is Reasonable.

b. Adequate Text-to-Background Contrast.

c. Font Size/Spacing is Easy to Read

d. Flash & Add-ons Are Used Sparingly

e. Images Have Appropriate ALT Tags

f. Site Has Custom Not-found/404 Page

5. Other user information for user help:

Like search option, sitemap, help files etc. Sitemap should be present with all the links in web sites with proper tree view of navigation. Check for all links on the sitemap.

"Search in the site" option will help users to find content pages they are looking for easily and quickly. These are all optional items and if present should be validated.

3.2.3 Critical factors need to consider while Functional testing of web Application

The following **factors need to be consider while testing functionality** of Website as **per the researcher recommendation**

1. Links

Objective is to check for all the links in the website.

All Internal Links

All External Links

All mail to links

Check for Broken Links

2. Forms

Test for the integrity of submission of all forms.

All Field Level Checks

All Field Level Validations.

Functionality of Create, Modify, Delete & View.

Handling of Wrong inputs (Both client & Server)

Default Values if any

Optional versus Mandatory fields.

3. Cookies

Cookies are small files stored on user machine. These are basically used to maintain the session mainly login sessions. Test the application by enabling or disabling the cookies in your browser options. Test if the cookies are encrypted before writing to user machine. If you are testing the session cookies (i.e. cookies expire after the sessions ends) check for login sessions and user stats after session end. Check effect on application security by deleting the cookies.

4. Web Indexing

Depending on how the site is designed using Meta tags, frames, HTML syntax, dynamically created pages, passwords or different languages, our site will be searchable in different ways.

Meta Tags

Frames

HTML syntax.

5. Database

Data consistency is very important in web application. Check for data integrity and errors while you edit, delete, modify the forms or do any DB related functionality. Check if all the database queries are executing correctly, data is retrieved correctly and also updated correctly.

6. **Validate your HTML/CSS:**

If you are optimizing your site for Search engines then HTML/CSS validation is very important. Mainly validate the site for HTML syntax errors. Check if site is crawlable to different search engines.

7. Data Verification and Validation

a. Is the Privacy Policy clearly defined and available for user access?
b. At no point of time the system should behave awkwardly when an invalid data is fed.
c. Check to see what happens if a user deletes cookies while in site.
d. Check to see what happens if a user deletes cookies after visiting a site

8. Data Integration

a. Check the maximum field lengths to ensure that there are no truncated characters?
b. If numeric fields accept negative values can these be stored correctly on the database and does it make sense for the field to accept negative numbers?
c. If a particular set of data is saved to the database check that each value gets saved fully to the database.(i.e.) Beware of truncation (of strings) and rounding of numeric values.

9. Date Field Checks

a. Assure that leap years are validated correctly & do not cause errors/miscalculations.

b. Assure that Feb. 28, 29, 30 are validated correctly & do not cause errors/ miscalculations.

c. Is copyright for all the sites includes Yahoo co-branded sites are updated

10. Numeric Fields

a. Assure that lowest and highest values are handled correctly.

b. Assure that numeric fields with a blank in position 1 are processed or reported as an error.

c. Assure that fields with a blank in the last position are processed or reported as an error an error.

d. Assure that both + and - values are correctly processed.

e. Assure that division by zero does not occur.

f. Include value zero in all calculations.

g. Assure that upper and lower values in ranges are handled correctly. (Using BVA)

11. Alphanumeric field Checks

a. Use blank and non-blank data.

b. Include lowest and highest values.

c. Include invalid characters & symbols.

d. Include valid characters.

e. Include data items with first position blank.

f. Include data items with last position blank.

12. Test business workflow-

This will include

a. Testing your end - to - end workflow/ business scenarios which takes the user through a series of webpage's to complete.

b. Test negative scenarios as well , such that when a user executes an unexpected step , appropriate error message or help is shown in your web application.

3.2.4 Critical factors need to consider while Security testing of web Application

The following **factors needs to consider while security testing of** web application as **per the researcher recommendation**

1. Secure Transmission

a. Check SSL versions, Algorithms, Key length

b. Check for Digital Certificate Validity (Duration, Signature and CN)

2. Data Validation

a. Test for Reflected Cross Site Scripting

b. Test for Stored Cross Site Scripting

Check the Web application for XSS (Cross site scripting). Any HTML, such as <HTML>, or any script such as <SCRIPT> should not be accepted by the application. If it is, the application can be prone to an attack by Cross Site Scripting.

3. Denial of Service

a. Test for anti-automation

b. Test for account lockout

4. Cryptography

a. Check if data which should be encrypted is not

b. Check for wrong algorithms usage depending on context

5. Risky Functionality – File Uploads

a. Test that acceptable file types are white listed

b. Test that file size limits, upload frequency and total file counts are defined and are enforced

6. Risky Functionality – Card Payment

a. Test whether card numbers are stored

7. HTML 5

a. Test Web Messaging

b. Test for Web Storage SQL injection

8. Information Gathering

a. Manually explore the site

b. Spider/crawl for missed or hidden content

9. Configuration Management

a. Check for commonly used application and administrative URLs

b. Check for old, backup and unreferenced files

10. Authentication

a. Test for user enumeration

b. Test for authentication bypass

11. Session Management

a. Establish how session management is handled in the application (eg. tokens in cookies, token in URL)

b. Check session tokens for cookies flags (http Only and secure)

12. Business Logic

a. Test for feature misuse

b. Test for lack of non-repudiation

13. Security Questions & Secret answer

a. Security question should be designed in such a fashion that they are not obvious to be known (For eg. What's your pet's name? Don't frame the question like this).It would be good if user is provided with option of choosing customized security question.

b. Secret / security answers should be stored in database as hashes and not plain text.

14. Captcha

a. Captcha characters should not be displayed in cyclic fashion.

b. Captcha images should not be allowed to download at one time using add on like Down Them All.

c. Use http://free-ocr.com/ to see if captcha could be deciphered Every refresh of a webpage should display new captcha every time.

d. Do not show the absolute path names of the captcha that is being displayed because it is easy to put assertions identifying the URL and then entering the according characters to pass the captcha.

e. Usage of question and answers type of captcha in textual format is good but, not good enough Researcher personally insists on using Google recaptcha for your web application because it has not been cracked till date. There are many captcha third party services out there but, researchers do not recommend those.

15. Password

a. Set of rules for setting a password should be same across all the modules like Registration form, Change password, and Forgot password. If these rules differ than hacker might exploit it through brute force method.

Example:

If the registration form does not validate for password minimum length as 8 chars but while changing password from user profile it validates for minimum length or vice versa. Now, as registration form accepts password which are less than 8 chars it becomes easy for hacker to apply brute force method.

b. Password enforcement of alphabets + numeric + special characters should be used in order to protect the account to a greater extent against brute force attack mechanisms.

16. Forgot your password

a. There need to be a restriction on number of forgot password requests sent per day or in "X" hours interval or have a captcha so that automated requests are not sent.

b. The URL has to expire on one use after being used to set new password.

c. The token associated with the URL should not be guessable or there should be any pattern which could be easily cracked.

d. If the URL is not used within "X" hours then it has to expire (Example: Once the URL is generated, if it is not used then it has to expire after "72 hours")

e. When new token is generated the old ones should expire even if they are not used. Example .com should not send the password via e- mails by resetting automatically

f. There has to be URL which should be used by end-user to set new password of his / her choice.

g. While typing secret answer in Forgot Password the secret answer needs to be masked (Secret Answer is also part of authentication which is similar to password, shoulder surfing or auto-complete stuff could be dangerous here compromising the end-user account).

h. Once the password is set, you might want to take end-user to logged in state or requesting him / her to login now with the hyperlink (I, personally would recommend taking to login page and requesting him / her to login with new **Change Password)**

i. Once the password is changed successfully. User should not be able to login again with his old password & new password both.

j. Login using the credentials on Mozilla Firefox ,Login with the same credentials on Google Chrome Now, change password for the account in

Google Chrome After this, refresh or try to navigate to some webpage which are allowed to be navigated only by logged in end-users Result: The end- user in Mozilla Firefox web browser has to log out as he / she is in the session which has old password

17. Secure Data Accessibility by direct pasting URL without Login

Test by pasting internal URL directly into browser address bar without login. Internal pages should not open.

3.2.5 Critical factors need to consider while Performance testing of web Application.

The following **factors needs to consider while performance testing of** web application as **per the researcher recommendation**

1. Connection speed

a. Try with different Connection speed
b. Time-out

2. Load Check/Measure the following:

a. What is the estimated number of users per time period and how will it be divided over the period?
b. Increase the number of users and keep the data constant.
c. Many users requesting a certain page at the same time or using the site simultaneously.
d. Does the home page load quickly? within 8 seconds.
e. Is load time appropriate to content, even on a slow dial-in connection?
f. Can the site sustain long periods of usage by multiple users?
g. Can the site sustain long periods of continuous usage by 1 user?
h. Is page loading performance acceptable over modems of different speeds?
i. Does the system meet its goals for response time, throughput, and availability?
j. Have you defined standards for response time (i.e. all screens should paint within 10 seconds)?
k. Does the system operate in the same way across different computer and network configurations, platforms and environments, with different mixes of other applications?

3. Stress

Stress testing is done in order to actually break a site or a certain feature to determine how the system reacts. Stress tests are designed to push and test system limitations and determine whether the system recovers gracefully from crashes. Hackers often stress systems by providing loads of wrong in-data until it crash and then gain access to it during start-up.

a. Typical areas to test are forms, logins or other information transaction components.
b. Performance of memory, CPU, file handling etc.
c. Error in software, hardware, memory errors (leakage, overwrite or pointers)

4. Continuous use

a. Is the application or certain features going to be used only during certain periods of time or will it be used continuously 24 hours a day 7 days a week?
b. Verify that the application is able to meet the requirements and does not run out of memory or disk space.

5. Volume

a. Can the site sustain large transactions without crashing?
b. Will the site allow for large orders without locking out inventory if the transaction is invalid?
c. Increase the data by having constant users.

3.2.6 Special Test Cases for Websites Heavily Loaded with AJAX and Flash

a. **Absence of an IDE for AJAX makes unit testing and debugging a discouraging task. Hence, excessive dependency is always there on the regression testing teams who need to factor this and design the test suite more elaborately and cover each single unit for all its possible functionalities.**
b. **High traffic Websites with AJAX will not work if JavaScript is disabled. Make sure that the user is prompted to enable JavaScript and even if the user doesn't do so, the site should not become totally inaccessible with JavaScript turned off.**

c. The 'Back' button on the browser may not work until developers provide that functionality. Testing teams need to ensure that this functionality is available on pages where there is a high probability of users hitting the 'Back' button.

d. It may be difficult to bookmark a particular section of an AJAX site. Testing teams must ensure that pages users may want to bookmark are free from this limitation.

e. The Flash based interface must be tested on different speeds such as modem, broadband, LAN etc.

f. The audio and video synchronization for flash movies and animations must be checked.

g. Finally, the Flash/AJAX application on the machine with minimum configuration must be tested.