MODULE 5: Testing for Specialized Environment

-by
Asst Prof Rohini Sawant

Testing Web-based Systems

- Web-based systems have had a great impact on our daily lives. These systems have evolved from small website add-ons to large multi-tiered applications.
- The computing environment for this rapidly growing web technology is complex as Internet is heterogeneous, distributed, and multi-platform. Besides the dynamic computing environment for web-based systems, user requirements for new features also add to the complexity in designing and testing these systems.
- Many attributes of quality web based systems such as ease of navigation, accessibility, scalability, maintainability, usability, compatibility and interoperability, security, readability, and reliability are not given due consideration during development.
- Therefore, proper testing of web based systems is needed in ensuring reliable, robust, and high-performing operation of web applications.
- The testing of web-based systems is a complex task not only due to the overwhelming number of users on the web but there is also a lot of difference between traditional systems and the web-based systems.

WEB-BASED SYSTEM

- The web-based software system consists of a set of web pages and components that interact to form a system which executes using web server(s), network,HTTP, and a browser, and in which user input affects the state of the system.
- Thus, web-based systems are typical software programs that operate on the Internet, interacting with the user through an Internet browser. Some other terms regarding these systems are given below:
- Web page is the information that can be viewed in a single browser window.
- A website is a collection of web pages and associated software components that are related semantically by content and syntactically through links and other control mechanisms. Websites can be dynamic and interactive.
- A web application is a program that runs as a whole or in part on one or more web servers and that can be run by users through a website. Web applications require the presence of web server in simple configurations and multiple servers in more complex settings. Such applications are called web based applications.
- Similar applications, which may operate independent of any servers and rely on operating system services to perform their functions, are termed web-enabled applications.
- These days, with the integration of technologies used for the development of such applications, there is a thin line separating web-based and web-enabled applications, so we collectively refer to both of them as web applications.

TRADITIONAL SOFTWARE AND WEB-BASED SOFTWARE

Web systems are based on client-server architecture wherein a client typically enables users to communicate with the server. Therefore, these systems share some characteristics of client-server architecture. However, there are a number of aspects of web systems that necessitate having different techniques to test them. These are discussed below:

1. Clients of the traditional client-server systems are platform-specific. This means that a client application is developed and tested for each supported client operating system. But the web client is operating within the web browser's environment. Web browsers already consist of operating system specific client software running on a client computer. But these browsers need to support HTML, as well as active contents to display web page information. For this purpose, browser vendors must create rendering engines and interpreters to translate and format HTML contents. In making these software components, various browsers and their releases introduce incompatibility issues.

2. In the traditional client-server systems, the normal flow of control is not affected by the user. But in web applications, users can break the normal control flow. For example, users can press the back or refresh button in the web browser.

TRADITIONAL SOFTWARE AND WEB-BASED SOFTWARE

3. Web-based systems have a more dynamic environment as compared to traditional client-server systems. In client-server systems, the roles of the clients and servers and their interactions are predefined and static as compared to web applications where client side programs and contents may be generated dynamically. Moreover, the environment for web applications is not predefined and is changing dynamically i.e. hardware and software are changing, configuration are ever-changing, etc. Web applications often are affected by these factors that may cause incompatibility and interoperability issues.

4. Due to the dynamic environment, web systems demand more frequent maintenance.

5. The user profile for web systems is very diverse as compared to client server systems. Therefore, the load on web access due to this diversity is not predictable.

CHALLENGES IN TESTING FOR WEB-BASED SOFTWARE

- **Diversity and complexity:** Web applications interact with many components that run on diverse hardware and software platforms. They are written in diverse languages and they are based on different programming approaches such as procedural, OO, and hybrid languages such as Java Server Pages (JSPs). The client side includes browsers, HTML, embedded scripting languages, and applets. The server side includes CGI, JSPs, Java Servlets, and .NET technologies. They all interact with diverse back-end engines and other components that are found on the web server or other servers.
- **Very short development time:** Clients of web-based systems impose very short development time, compared to other software or information systems projects. For e.g. an e-business system, sports website, etc.
- **Continuous evolution:** Demand for more functionality and capacity after the system has been designed and deployed to meet the extended scope and demands, i.e. scalability issues.

CHALLENGES IN TESTING FOR WEB-BASED SOFTWARE

- **Dynamic environment:** The key aspect of web applications is its dynamic nature. The dynamic aspects are caused by uncertainty in the program behaviour, changes in application requirements, rapidly evolving web technology itself, and other factors. The dynamic nature of web software creates challenges for the analysis, testing, and maintenance for these systems. For example, it is difficult to determine statically the application's control flow because the control flow is highly dependent on user input and sometimes in terms of trends in user behaviour over time or user location. Not knowing which page an application is likely to display hinders statically modeling the control flow with accuracy and efficiency.
- **Compatibility and interoperability:** As discussed above, there may also be compatibility issues that make web testing a difficult task. Web applications often are affected by factors that may cause incompatibility and interoperability issues. The problem of incompatibility may exist on both the client as well as the server side.

TESTING OF WEB-BASED SYSTEMS

- Web-based systems have a different nature as compared to traditional systems.
- We have seen the key differences between them.
- Due to the environment difference and challenges of dynamic behaviour, complexity and diversity also makes the testing of these systems a challenge.
- These systems need to be tested not only to check whether it does what it is designed to do but also to evaluate how well it appears on the (different) web browsers.
- Moreover, they need to be tested for various quality parameters which are a must for these systems like security, usability, etc. Hence, a lot of effort is required for test planning and test designing.
- The strategy for testing web-based systems is the same as for other systems, i.e. verification and validation.
- Verification largely considers the checking of analysis and design models which have been described above. Various types of testing derive from the design models only. However, the quality parameters are also important factors which form the other types of testing

INTERFACE TESTING

- Interface is a major requirement in any web application. More importantly, the user interface with web application must be proper and flexible.
- Therefore as a part of verification, present model and web scenarios model must be checked to ensure all interfaces.
- The interfaces between the concerned client and servers should also be considered. There are two main interfaces on the server side: web server and application server interface and application server and database server interface
- Thus, in interface testing, all interfaces are checked such that all the interactions between these servers are executed properly.
- Errors are handled properly. If database or web server returns any error message for any query by the application server, then the application server should catch and display these error messages appropriately to users. Check what happens if the user interrupts any transaction in between? Check what happens if connection to web server is reset in between? Compatibility of server with software, hardware, network, and database should also be tested.

USABILITY TESTING

- The actual user of application should feel good while using the application and understand everything visible to him on it. Usability testing is not a functionality testing, but the web application is reviewed and tested from a user's viewpoint.
- The importance of usability testing can be realized with the fact that we can even lose users because of a poor design.
- For example, check that form controls, such as boxes and buttons, are easy to use, appropriate to the task, and provide easy navigation for the user.
- The critical point for designers and testers in this attesting is that web application must be as pleasant and flexible as possible to the user.
- Usability testing may include tests for navigation. It refers to how the user navigates the web pages and uses the links to move to different pages.
- Besides this, content should be logical and easy to understand.
- Check for spelling errors. Use of dark colours annoy users and should not be used in the site theme.
- You can follow some standards that are used for web page and content building.
- Content should be meaningful.
- All the anchor text links should work properly.
- Images should be placed properly with the proper sizes.

The general guide Usability Testing are:

1. Present information in a natural and logical order.
2. Indicate similar concepts through identical terminology and graphics. Adhere to uniform conventions for layout, formatting, typefaces, labeling, etc.
3. Do not force users to remember key information across documents.
4. Keep in consideration that users may be from diverse categories with various goals. Provide understandable instructions where useful. Lay out screens in such a manner that frequently accessed information is easily found.
5. The user should not get irritated while navigating through the web application. Create visually pleasing displays. Eliminate information which is irrelevant or distracting.
6. Content writer should not mix the topics of information. There should be clarity in the information being displayed.
7. Organize information hierarchically, with more general information appearing before more specific detail. Encourage the user to delve as deeply as needed, but to stop whenever sufficient information has been received.
8. Check that the links are active such that there are no erroneous or misleading links.

CONTENT TESTING

- The content we see on the web pages has a strong impression on its user.
- If these contents are not satisfactory to him, he may not visit the web page again.
- Check the completeness and correctness properties of web application content.
- Check that certain information is available on a given web page, links between pages exist, or even check the existence of the web pages themselves (completeness property).
- Furthermore, web application content may need to be checked against semantic conditions to see if they meet the web document (correctness property). Therefore the contents should be correct, visible, flexible to use, organized, and consistent.

Static testing may consider checking the following points:

1. Various layouts.
2. Check forms for their field validation, error message for wrong input, optional and mandatory fields with specified length, buttons on the form, etc.
3. A table is present and has the expected number of rows and columns and pre-defined properties.
4. Grammatical mistakes in text description of web page.
5. Typographical mistakes.
6. Content organization.
7. Content consistency.
8. Data integrity and errors while you edit, delete, modify the forms.
9. Content accuracy and completeness.
10. Relationship between content objects.
11. Text contents.
12. Text fragments against formatting expectations. This differs slightly from simple text checking in that the formatting tags can be located loosely on the page as opposed to a fixed string for text content.
13. Graphics content with proper visibility.
14. Media contents to be placed at appropriate places.
15. All types of navigation links like internal links, external links, mail links, broken links to be placed at appropriate places.
16. All links on a web page are active.

NAVIGATION TESTING

We have checked the navigation contents in interface testing. But to ensure the functioning of correct sequence of those navigations, navigation testing is performed on various possible paths in the web application. Design the test cases such that the following navigations are correctly executing:

1. Internal links

2. External links

3. Redirected links

4. Navigation for searching inside the web application

The errors must be checked during navigation testing for the following:

1. The links should not be broken due to any reason.

2. The redirected links should be with proper messages displayed to the user.

3. Check that all possible navigation paths are active.

4. Check that all possible navigation paths are relevant.

5. Check the navigations for the back and forward buttons, whether they are working properly.

CONFIGURATION/COMPATIBILITY TESTING

Some points to be careful about while testing configuration are:

1. There are a number of different browsers and browser options. The web application has to be designed to be compatible for majority of the browsers.
2. The graphics and other objects on a website have to be tested on multiple browsers. If more than one browser will be supported, then the graphics have to be visually checked for differences in the physical appearance. Some of the things to check are centering of objects, table layouts, colours, monitor resolution, forms, and buttons.
3. The code that executes from the browser also has to be tested. There are different versions of HTML. They are similar in some ways but they have different tags which may produce different features. Some of the other codes to be tested are Java, JavaScript, ActiveX, VBscripts, Cgi-Bin Scripts, and Database access. Cgi-Bin Scripts have to be checked for end-to-end operations and is most essential for e-commerce sites. The same goes for database access.
4. All new technologies used in the web development like graphics designs, interface calls like different API's, may not be available in all the operating systems. Test your web application on different operating systems like Windows, Unix, MAC, Linux, Solaris with different OS flavors.

SECURITY TESTING

Today, the web applications store more vital data and the number of transactions on the web has increased tremendously with the increasing number of users. Therefore, in the Internet environment, the most challenging issue is to protect the web applications from hackers, crackers, spoofers, virus launchers, etc. Through security testing, we try to ensure that data on the web applications remain confidential, i.e. there is no unauthorized access. Security testing also ensures that users can perform only those tasks that they are authorized to perform.

In a web application, the risk of attack is multifold, i.e. it can be on the web software, client-side environment, network communications, and server-side environments. Therefore, web application security is particularly important because these are generally accessible to more users than the desktop applications. Often, they are accessed from different locations, using different systems and browsers, exposing them to different security issues, especially external attacks. It means that web applications must be designed and developed such that they are able to nullify any attack from outside. Therefore, this issue is also related to testing of web application in terms of security. We need to design the test cases such that the application passes the security test.

PERFORMANCE TESTING

- Web applications' performance is also a big issue in today's busy Internet environment.
- The user wants to retrieve the information as soon as possible without any delay.
- This assumes the high importance of performance testing of web applications. Is application able to respond in a timely manner or is it ready to take the maximum load or beyond that? These questions imply that web applications must also be tested for performance.
- Performance testing helps the developer to identify the bottlenecks in the system and can be rectified.
- In performance testing, we evaluate metrics like response time, throughput, and resource utilization against desired values.
- Using the results of this evaluation, we are able to predict whether the software is in a condition to be released or requires improvement before it is released.
- Moreover, we can also find the bottlenecks in the web application. Bottlenecks for web applications can be code, database, network, peripheral devices, etc

PERFORMANCE TESTING

Performance Parameters

Performance parameters, against which the testing can be performed, are given here:

Resource utilization The percentage of time a resource (CPU, Memory, I/O, Peripheral, Network) is busy.

Throughput The number of event responses that have been completed over a given interval of time.

Response time The time lapsed between a request and its reply.

Database load The number of times database is accessed by web application over a given interval of time.

Scalability The ability of an application to handle additional workload, without adversely affecting performance, by adding resources such as processor, memory, and storage capacity.

Round-trip time How long does the entire user-requested transaction take, including connection and processing time?

AGILE TESTING

- Agile systems are designed to be quick and easy to use. Much effort has been expended on the development of agile software development processes.
- Unfortunately, those software development processes have not placed much emphasis on software testing, even though in many systems, testing consumes about one-half the total developmental resources.
- Effective agile processes have the following three characteristics:
- ■■ Well-defined objectives
- ■■ Open information/communication channels
- ■■ Task-performance flexibility
- The best way to build an agile software testing process is to empower the actual testers to build one. Software testers can build an agile testing process by focusing on the following three goals:
- ■■ Eliminating non productive aspects of testing
- ■■ Identifying and incorporating best testing practices
- ■■ Building the process using the practices that offer the greatest payback with the highest probability of successful implementation

AGILE TESTING

The agile implementation team can be as few as three or as many as five people, but each person should be a stakeholder in the testing. At a minimum, the team should have these representative members:

■■ Software manager

■■ Development representative, particularly, a project leader who respects the importance of testing

■■ User representative

- Agile testing is unstructured as compared to the waterfall approach and there is minimal planning.
- Agile testing is well suited for small projects.
- As testing begins at the start of the project, errors can be fixed in the middle of the project.
- There is very less documentation required for agile testing
- In this approach, every iteration has its own testing phase. The regression tests can be run every time new functions or logic are released.
- In agile testing shippable features of the product are delivered to the customer at the end of an iteration.
- Testers and developers work closely in Agile testing.

Agile Testing Principles

There are some principles of agile testing process which are given below:

Testing is continuous: Agile team tests continuously because it is the only way to ensure continuous progress of the product.

Continuous feedback- Agile testing provides feedback on an ongoing basis and this is how your product meets the business needs.

Tests performed by the whole team: In a traditional software development life cycle, only the test team is responsible for testing but in agile testing, the developers and the business analysts also test the application.

Decrease time of feedback response: The business team is involved in each iteration in agile testing & continuous feedback shortens the time of feedback response.

Simplified & clean code: All the defects which are raised by the agile team are fixed within the same iteration and it helps in keeping the code clean and simplified.

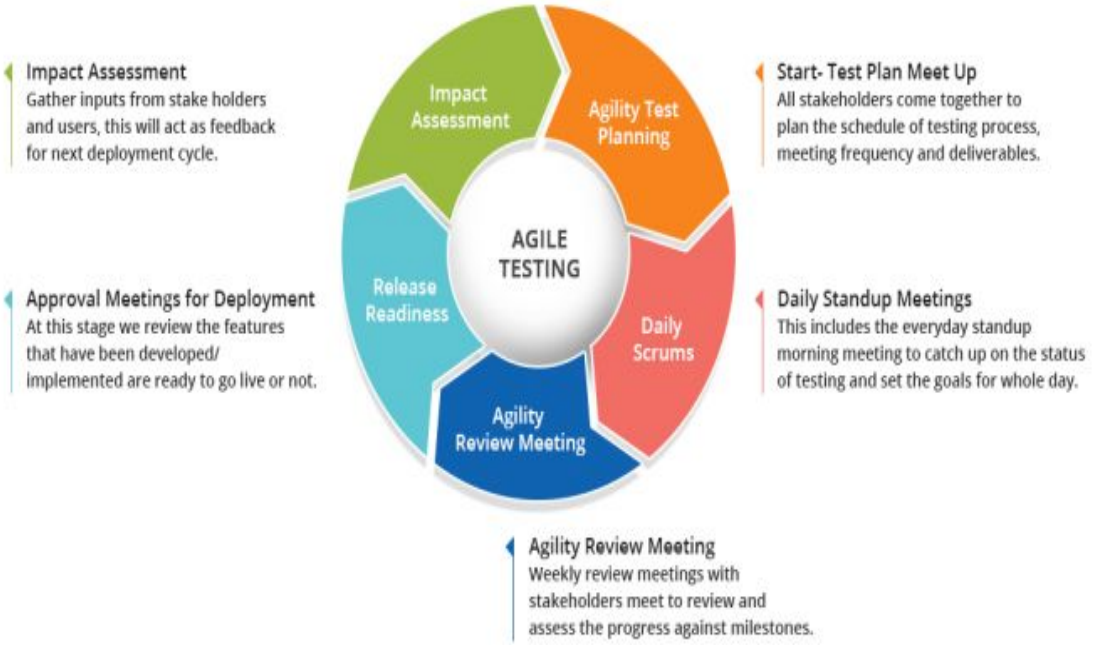
Less documentation: Agile teams use a reusable checklist, the team focuses on the test instead of the incidental details.

Test Driven: In agile methods, testing is performed at the time of implementation whereas, in the traditional process, the testing is performed after implementation.

Advantages of Agile Testing Methodology

The benefits of the agile testing approach are as follows:

- It saves time and money
- Agile testing reduces documentation
- It is flexible and highly adaptable to changes
- It provides a way for receiving regular feedback from the end user
- Better determination of issues through daily meetings



Challenges in Agile Testing

1. Changing Requirements

It can sometimes happen that management changes requirements or drops stories during a sprint, even though this is not encouraged in an agile/Scrum framework. This means that work already half-done needs to be discarded or modified, which changes the scope of testing unexpectedly.

2. Not Enough Information

Product owners, who are responsible for developing user stories, may have an idea about a new feature but may not be aware of the specifics. This means they can't write a good set of acceptance criteria. If there is missing information about requirements, testers can't build comprehensive test cases.

3. Continuous Testing

Testing is not restricted to one part of the development process, rather it's an ongoing activity that starts before the development phase. This creates a major challenge because testers are expected to start building tests for features before coding has even started, or while coding is taking place.

4. Frequent Regression Cycles

Developers frequently and continuously add features to the product. This can cause regressions in previous features. Testers use regression tests to identify this problem and overcome it, but manual regression testing is impractical in a fast-paced agile environment.

Challenges in Agile Testing

5. Lack of Communication

If communication between developers, testers and the product owners is lacking, agile testing will simply not work.

6. No Quality Measurement

Agile teams today have no single measurement of quality, which they can use to optimize and plan testing efforts. Most agile teams are flying blind, responding to production failures or bugs, but unable to proactively focus on product areas which have the biggest quality issues.

7. Tester’s Availability

Sometimes, the testers don’t possess the relevant skills to perform API testing or Integration testing. This might result in skipping some important tests. You can either choose testing software that does not require coding for testers or you can train the testing team to carry out all the essential tests effectively.

SCRUM

- **Scrum in Software Testing** is a methodology for building complex software applications. It provides easy solutions for executing complicated tasks.
- **Scrum Testing** is a testing done in scrum methodology to verify the software application requirements are met. It involves checking non-functional parameters like security, usability, performance etc.
- There is no active role of tester in the process so it is usually performed by developers with Unit Test. Sometimes dedicated test teams are needed depending on nature & complexity of project.

Following are Key Features of Scrum-

- Scrum has a short fixed schedule of release cycles with adjustable scope known as **sprints** to address rapidly changing development needs. Each release could have multiple sprints. Each Scrum Project could have multiple Release Cycles.
- A repeating sequence of **meetings, events, and milestones**
- A practice of testing and implementing new requirements, known as **stories**, to make sure some work is released ready after each sprint

ROLES IN SCRUM

Product Owner	Scrum Master	The Team
He/She defines features of the product.	He/She manages the team and look after the team's productivity	The team is usually about 5-9 members
Product Owner decides the release date and corresponding features	He/She maintains the block list and removes barriers in the development	It includes developers, designer and sometimes testers, etc.
They prioritize the features according to the market value and profitability of the product	He/She coordinates with all roles and functions	The team organizes and schedule their work on their own
He/She is responsible for the profitability of the product	He/She shields team from external interferences	Has right to do everything within the boundaries of the project to meet the sprint goal
He/She can accept or reject work item result	Invites to the daily scrum, sprint review and planning meetings	Actively participate in daily ceremonies

SCRUM ARTIFACTS

A scrum process includes

- **User stories:** They are a short explanation of functionalities of the system under test. Example for Insurance Provider is – “Premium can be paid using the online system.”
- **Product Backlog:** It is a collection of user stories captured for a scrum product. **The product owner prepares** and maintains the product backlog. It is prioritized by the product owner, and anyone can add to it with approval from the product owner.
- **Release Backlog:** A release is a time frame in which the number of iterations is completed. **The product owner co-ordinates** with the scrum master to decide which stories should be targeted for a release. Stories in the release backlog are targeted to be completed in a release.
- **Sprints:** It is a set period of time to complete the user stories, decided by the product owner and developer team, usually 2-4 weeks of time.
- **Sprint Backlog:** It’s a set of user stories to be completed in a sprint. During sprint backlog, work is never assigned, and the team signs up for work on their own. It is owned and managed by the team while the estimated work remaining is updated daily. It is the list of task that has to be performed in Sprint
- **Block List:** It is a list of blocks and unmade decisions owned by scrum master and updated daily
- **Burndown chart:** Burn-down chart represents overall progress of the work in progress and work completed throughout the process. It represents in a graph format the stories and features not completed

PROCESSES IN SCRUM

- **Sprint Planning:** A sprint begins with the team importing stories from the release backlog into the sprint backlog; it is hosted by scrum master. The Testers estimate effort to test the various stories in the Sprint Backlog.
- **Daily Scrum:** It is hosted by scrum master, it last about 15 minutes. During Daily Scrum, the members will discuss the work completed the previous day, the planned work for the next day and issues faced during a sprint. During daily stand-up meeting team progress is tracked.
- **Sprint Review/ Retrospective:** It is also hosted by scrum master, it last about 2-4 hours and discuss what the team has accomplished in the last sprint and what lessons were learned.