USABILITY TESTING FOR THE WEB

VIKRAM V. INGLESHWAR, YAHOO!

Today's sophisticated Web applications make tracking and listening to users more important than ever.

Today's Internet user has more choices than ever before, with many competing sites offering similar services. This proliferation of options provides ample opportunity for users to explore different sites and find out which one best suits their needs for any particular service. Users are further served by the latest generation of Web technologies and services, commonly dubbed Web 2.0, which enables a better, more personalized user experience and encourages user-generated content.

Although there is considerable debate over the definition of Web 2.0 (and much criticism of it being merely a marketing buzzword), the term is useful in distinguishing key innovations, such as weblogs, social bookmarking, tagging, wikis, RSS feeds, Ajax, Web APIs, and online Web services, that have significantly altered the Web user experience since the 1990s.

As the Web matures and evolves, users increasingly view it as a platform for applications, and they demand a much greater degree of personalization and control. As such, it's important for Web developers to discover and closely track certain user feedback, such as their motives for coming to a particular site, areas of the site they find difficult and/or confusing, and the features they like the most. Retaining existing users, encouraging them to spend more time on the site, and bringing in new users have become greater challenges than ever. Usability testing is a way of addressing these challenges.

When the Internet was in its nascent phase, it was viewed as merely one of many sources for getting information. As PC and Internet use has increased across the globe, users have become more savvy and know more precisely what kind of information they are looking for on the Web and how they would like to receive it. This has made usability testing an essential part of the testing cycle. Unfortunately, its importance is often overlooked, to the detriment of any particular Web application. We need to change our mind-set and adopt usability testing as a part of the everyday testing cycle.

This article looks at the challenges of usability testing, particularly within the context of the current wave of Web 2.0 technologies. It does not necessarily present solutions, but instead highlights most of the challenges, issues, and difficulties you might face in your own projects.

OBJECTIVES OF USABILITY TESTING

Usability can be defined as the degree to which a given piece of software assists the person sitting at the keyboard in accomplishing a task, as opposed to becoming an impediment. Usable systems are most often assessed according to these three criteria:

- Ease of learning and learning retention
- Speed of task completion
- Error rate and subjective user satisfaction/quality of service

There are a few basic principles to consider, based on these criteria:

First and foremost, make your application easy to use. This will reduce user churn, prevent calls from frustrated customers, and help keep people on your site. But what is the benchmark that says that a Web application can be easily used by anyone who visits your site? This is a tricky question, because what's easy for some people might be difficult for others (just as integration in mathematics is simple for some people and a sleep killer for others). So the question is, how can you know if your Web application is simple to understand or if users have to go through the help pages to find out exactly how things work? One simple technique is to give your application to the QA team to test without permitting them to read the functional spec or product requirement document. You needn't remove the help pages entirely, but users shouldn't be forced to consult the help pages often; and on each subsequent visit to a site they should be able to remember whatever they learned during their first time using the application. If they have to refer to help pages

USABILITY TESTING FOR THE WEB

each time they visit your application, then there is something truly wrong and you need to take corrective action.

Also, many sites expose their APIs to developers, who then add this functionality to existing applications. Do users feel comfortable using these new features? A good site will make users feel comfortable learning new features and inspired to explore the site further.

Make your application fast. *Fast* is a relative term; you should always make sure that the speed of your application is on par with that of your competitors. To increase the responsiveness of services, many sites now use Ajax technology, which makes Web applications behave more like desktop applications. This is an intermediate solution between the slower response times provided by Web servers and the normally faster response times of desktop applications.

Another technique to increase responsiveness is to overlap or clump together operations without making things look too messy. Find out different permutations and combinations of performing an operation in your application, then analyze which repetitive tasks could be overlapped or achieved in a single step (aka, refactoring). One way of doing this is to use Web server logs to find out which pages users visit most and which pages they find difficult to use. For example, suppose you have an e-commerce site that leads users through a six-step (page) process. If the Web server log shows that many users are leaving the site during page four, chances are that page has a problem. Perhaps the user is finding it tedious to input so much information, or perhaps the server process itself is failing. Mining the Web server logs is a good technique to track user behavior closely, but it's a tedious, time-consuming task. Later in the article I discuss *multivariable testing*, which can help resolve this issue.

Listen to your users. Many times I have reported issues to sites and failed to receive any acknowledgment. If you want these loyal customers to keep giving valuable feedback, you should immediately prioritize the issue and get it resolved in a timely manner. Allowing users to send feedback easily and making them feel that you are listening to them is very important.

You can also provide a very Web 2.0 user-feedback feature on your site, where users can comment on the site's features and discuss them with others. This is very good way to find your users' pulses and discover what you are doing, both right and wrong.

HOW TO ACHIEVE BETTER USABILITY

The best way to get usability feedback is to test the application with a QA group that knows nothing about it. Once testers have gone through the product requirements document, the functional spec, and at last, the testing spec, they will be testing the application with a specific mind-set influenced by their prior exposure to it. It is difficult for them to break this mind-set and test while "thinking outside the box."

In one of my own projects, we had a dedicated QA team testing a product for nearly three months. In between this testing cycle we asked another QA team to jump in and do the ad hoc testing; this helped unearth many usability issues. An engineer who is totally unaware of an application can test it as an end user and may come up with valuable insights and observations.

Checking the usability of a piece of functionality is as important as testing that piece of functionality itself. You need to discover what challenges and difficulties users may encounter with your application and how you can handle those in advance. Make notes of such points and discuss them during periodic team meetings; if a point is valid it should be communicated to the project manager. Then the manager and the QA team can discuss these concerns and resolve the problem. The accompanying sidebar outlines some basic techniques for unearthing many of these usability problems.

TRACKING USER BEHAVIOR

In the past, most sites acted strictly as information providers, and developers hardly bothered to track users. Now, with more and more users treating the Web as a platform, it's imperative that developers pay more attention to how they use certain sites and what their preferences are, and respond to that information accordingly.

One of the techniques used to track user behavior is called *multivariable testing*. Generally speaking, this is a series of tests, each representing an area on a Web page where content can be rotated through. Multivariable testing allows you to track user behavior at runtime and to make changes and then check how they're working immediately. Many vendors make products to assist in multivariable testing, but there is still the challenge of fulfilling user needs based on their geographic locations, user behavioral patterns, and so on. For example, the needs of users in the United States are different from

Usability Testing Guidelines and Standards

1. Create a feature usability matrix.
List all the functions in your application and ask people outside your team to test all of them. Testers should note their observations for each function and rate them according to their ease of use and practical utility.

Using this data, prepare a *feature usability matrix*, which describes the application's functionality, how easy it is to understand, and whether users require external help with any particular feature. This matrix will help you discover which functionality performs well and which needs rework, and it will help you to continuously improve the site's usability.

2. Create a comparison usability matrix.
Look at the features list for your site. Check how these features are implemented in competing Web sites. Record points of comparison, such as ease of use, look-and-feel, responsiveness, etc.

With these observations, prepare a *comparison usability matrix* and have this reviewed by your QA team. Whichever features get a lower rating against your competitors should be corrected. This matrix will ensure that your services are on par with your competitors.

3. Create a customer feedback usability matrix.
The automobile industry is attentive to customer feedback and tries to implement common change requests in their new models. Toyota's success owes a lot to its customer feedback processes. The same best practices apply to Web applications. Can customers provide feedback easily on your site? Can they easily find existing issues discovered by others? What process is in place to take action on this feedback?

Providing a rating/review feature such as Digg.com on your site will help improve the product and build strong customer relationships. You can then use this data to create a *customer feedback usability matrix* to track how responsive you are to customer feedback.

those of users in Korea. If you are trying to reach a global audience, you need to take this into consideration.

Tracking users with cookies is a tricky proposition; most users will feel like somebody is spying on them. Using Web 2.0 technologies, we might one day be able to track user behavior without users' knowledge and without putting cookies on their machines. This would be similar to vehicular movement studies on highways for proper traffic management. In these studies, cameras are placed at regular intervals to monitor traffic density at particular times of the day, which types of vehicles are moving by that road, and other variables. If we come up with similar technology that can monitor people's behavior on a particular Web site, we can find out where people are coming from, their browsing patterns, etc.

Unless we conduct a survey, however, finding out the age group, sex, nationality, etc. of users will be the main challenge. Even a survey will not guarantee that users will provide accurate data, but we can make educated guesses about them. For example, if a user is spending a lot of time on education sites that discuss engineering degrees, we might guess that this user falls into the 20–35 age group.

WHAT'S NEXT?

M-Web 2.0 is my concept of the Web a couple of years down the line. The user base of mobile and handheld devices is increasing much faster than anyone can imagine. As cities are developing wireless networks and new software and features are being added to mobile devices, I can imagine 95 percent of the population using WAP-enabled mobile devices in the next few years. The limitation with PCs and laptops is that they are bulky and the user has to log in to connect to the Web. Keeping this future trend in mind should ensure mobile browser support for many Web applications.

Are we ready to adopt this change? How are we going to shift billions of Web sites to mobile browsers without losing usability? This is the next big challenge that lies ahead of us. Q

ADDITIONAL RESOURCES

Usability Testing of World Wide Web Sites; http://stats.bls.gov/ore/htm_papers/st960150.htm.
Internet World Stats: Usage and Population Statistics; http://www.internetworldstats.com/stats.htm.

LOVE IT, HATE IT? LET US KNOW
feedback@acmqueue.com or www.acmqueue.com/forums

VIKRAM V. INGLESHWAR is a QA engineer for Yahoo! in Bangalore, India. He has spent more than four years in the IT industry, testing PDA and telecom software and Web-based applications. He holds a bachelor of engineering degree and has completed a Diploma in Advanced Computing course. His areas of interest are automation and white-box testing.