

Knowledge Quest

Vol. 33, no. 3

Title: Testing Library Websites for Usability

Authors: Cem Kaner and Rebecca Fiedler

Author: ID: Cem Kaner is a professor of software engineering at Florida Institute of Technology. Rebecca Fiedler is a Ph.D. student in Instructional Technology at the University of Central Florida.

Work: Florida Institute of Technology, 150 West University Boulevard, Melbourne, FL 32901, 321-674-7137, 321-727-8084, kaner@kaner.com; University of Central Florida, 4000 Central Florida Blvd, Orlando, FL 32826, 321-230-2615, 321-727-8084, becky@msfiedler.com

**Home: Cem Kaner, 1600 Seabury Point Road, Palm Bay, FL 32907; Rebecca Fiedler, 1005 Wainright Drive, Oviedo, FL 32765**

## Testing Websites

Your Web site visitor faces a complex challenge. Using a particular web browser (e.g., Internet Explorer, Mozilla Firefox, Opera, Netscape) on either a Mac or PC, your patron accesses your library's web server through the school's network or Internet service provider (ISP). From there, the user issues a request, perhaps for a Web page you created with your hours of operation, a link to a tutorial hosted by another institution in a different time zone, or a query to one of your subscription databases. Each transaction is complex requiring all equipment, software, and databases to interact seamlessly so that your user's request can be fulfilled.(1)

Web testing is a process of technical investigation - of imagining and hunting down the potential problems or risks that your Web site visitor faces. *Compatibility testing* looks at the interaction among browsers, computers, operating systems, printers, and other hardware or software that can cause applications to fail. *Security testing* investigates unauthorized access to your equipment or data for the purpose of using or changing your computer hardware or software. *Functional testing* looks for broken features or functions. The focus of this article is *usability testing* which asks whether the user will find the site difficult, unpleasant, or inefficient to navigate.

## Start with Basic Functional Testing

If your site fails to work it is unusable. When the user makes a selection or enters data, the software should accept correct inputs and reject everything else, perhaps with a polite and clear error message. Our Black Box Testing Course (<http://www.testingeducation.org/k04/index.html>) describes how to do this. Alternatively, there are several good introductory books. (2)

Link testing is another key checking task, since links go out of date quickly.(3) Xenu's Link Sleuth <<http://home.snafu.de/tilman/xenulink.html>> or other free or low-cost link-checkers (4) will expose links that point to a no-longer-existent page. Another critical problem is link substitution, in which a new organization buys the domain name previously owned by another. A few years ago, <[www.4cite.org](http://www.4cite.org)> was the home page for Americans for Fair Electronic Commerce Transactions to which ALA belonged. The AFFECT coalition changed it's URL <<http://www.ucita.com>> but old links to a page focused on the Uniform Computer Information Transactions Act now reach a collection of sexually explicit images. Needless to say, periodic checking of every link on your site is very important.

## User Interfaces, Responses, Subpopulations, and User Scenarios

When programmers perform usability testing, they often combine user interface testing and user response testing.

*User interface testing* checks every pull-down menu, button, link and element. The tester enters, erases and edits data in every field, shows and hides every toolbar, opens and closes windows, and moves them around.

*User response testing*, or simply user testing, involves putting the product in the hands of users and watching what happens. A tester in a well-equipped usability laboratory may assign specific tasks to the user and record the user's keystrokes and mouse clicks,(5) or videotape the user and the computer screen, or watch the user through one-way mirror-windows. (6) At the other end of the spectrum the tester distributes "beta" copies (acknowledged to still contain problems or bugs) of the software to outsiders who use the product and give more or less detailed feedback.(7)

A school library media specialist might conduct a formal user response test by asking patrons to compare two web pages that provide access to the same data, watching how they approach the task and collecting user satisfaction data.(8) You can locate two alternative designs through School Libraries on the Web <<http://www.sldirectory.com>>. Pick two sites that illustrate different potential designs for your site or different solutions to a specific problem that you're trying to solve. In order to determine which approach is better, allow users to do whatever they want with whichever site, then report to you about their experiences. Alternately, you can ask them to focus on a specific task such as a search or retrieval of information, give a written assignment that includes the task and observe how the users do that same task at each site. In evaluating their performance, you might measure such variables as how long it took them to complete each task, how many completed it, how often they made mistakes, how long it took them to recover from errors, how often they looked for help, or how often they expressed dissatisfaction. (9) If you don't have that much time to devote to the observation, you can assign specific tasks and measure student performance and attitude at only one site you might replicate. While you won't be able to judge that "this is better than that," you will probably recognize some aspect of the design that can be improved.

Done formally, user response testing is slow and expensive. Steve Krug (10) offers several recommendations for running cheaper, faster versions of the tests to obtain most of the information at a much lower cost.

*Usage tests* consider the statistical patterns of use over time across a large pool of people. The result is often called an *operational profile*. Ideally you learn how often people do what, in what sequence.(11) Since usage tests take a lot of work to develop and are intended to collect extensive records of use about Web site whose design is stable, (12) such tests are not feasible for initial development of a small new site or for fast incremental revisions.

Different people have different needs, constraints and preferences, but *user subpopulations* are often insufficiently considered. Users with special needs are of particular interest to both school and public libraries, especially now that Section 508 requires Federal agencies' technology to be accessible to individuals with handicaps. (13) While our focus is on testing rather than design, U.S. Department of Health and Human Services web site <<http://www.usability.gov>> provides a large collection of design guidelines and usability resources. There are some useful tools which

determine whether your site is likely to work for people with special needs. For example, Bobby <bobby.watchfire.com> tests for accessibility and creates a report, STEP 508 <www.section508.gov/index.cfm?fuseaction=content&id=155> helps you prioritize the accessibility problems you find and Vischeck <www.vischeck.com> shows web pages and images as someone who is colorblind might see them.

### **How to Create a User Scenario**

While reading this issue, you've probably thought of many different scenarios—or stories—about how different people might use your new or revamped Website for different purposes. These are *user scenarios* (14). An important variation on this theme is the *use case* (15) which lays out the user's goal and sequence of work, but not the user's motivation. While scenarios are normally the tools of system designers, (16) testers can base most of their investigations on scenarios created by the designers (17) or on scenarios developed by the testers themselves. From Kaner (18) and the design texts in our references, we've culled some suggestions for brainstorming scenarios.

Make a list of the different types of users of your system. Don't worry about capturing them all. You'll think of others and revise the lists and tests later.

For each type of user, consider their interests. Why are they using this system? What do they want from it? Then make a determination: are these interests favored (the system should help them), disfavored (it should prevent them) or neutral (you don't care whether they can use it that way or not, but you aren't going to intentionally enable that use)?

For each favored use, write a short story about a person trying to use your Web site for this interest. Why do they want to do this? What do they know, or not know that they might need to know, to use your Web page? What experiences have they had with other systems (on paper or with other software) that might lead them to try to use your Web page in a certain way? What do they do? What is supposed to happen?

Write analogous stories for the disfavored uses. What do you think they would try, and how well does the system block them?

Now, run the tests. Does your Web site provide the benefits it is supposed to provide? Did it get in the way at any point? Confuse you? Lead you down the wrong path? Annoy you with inappropriate or cryptic messages? Did it provide the output you expected? Did you enjoy using it?

### **In Closing**

Usability testing can not replace good usability design, but it can reveal errors in implementation. Usability testing can consume a lot of time and a large budget, or it can be done on a shoestring. Since a useful and usable website will contribute to the library media center credibility and mission, we encourage you to take the time to do usability testing for your school library media center's Web site.

[1] Hung Q. Nguyen, Bob Johnson and Michael Hackett, *Testing Applications on the Web*, 2<sup>nd</sup> edition (New York: John Wiley & Sons, 2003).

- [2] Cem Kaner, Jack Falk and Hung Q. Nguyen, *Testing Computer Software*, 2<sup>nd</sup> edition, (New York: John Wiley & Sons, 2003). Louise Tamres, *Introducing Software Testing*, (London: Addison-Wesley, 2002).
- [3] Wallace Koehler, "A Longitudinal Study of Web Pages Continued: A Consideration of Document Persistence," January 2004, *Information Research* 9, no. 2 (2004) <<http://informationr.net/ir/9-2/paper174.html>> (13 August, 2004).
- [4] Tucows, "Web Building Tools – Validators," <[http://www.tucows.com/htmlval95\\_default.html](http://www.tucows.com/htmlval95_default.html)> for Windows or "Macintosh – Web Building Tools – Validators," <[http://mac.tucows.com/webbuilding\\_validators\\_default.html](http://mac.tucows.com/webbuilding_validators_default.html)> or Download.com <http://www.download.com/> (13 August, 2004).
- [5] Spector, "Spector and eBlaster Spy Software – Internet Monitoring Software," [www.spectorsoft.com](http://www.spectorsoft.com) or Easy QA, "Easy QA- Functional and GUI Tools," <http://www.easy-qa.com/pages/easy-qa-tools-Functional-GUI-Testing-Tools.htm> (13 August, 2004).
- [6] Jeffrey Rubin, *Handbook of Usability Testing*, (New York: John Wiley & Sons, 1994).
- [7] Michael R. Fine, *Beta Testing for Better Software*, (New York: John Wiley & Sons, 2002).
- [8] Mark Notes, "Three Looks at Users: A Comparison of Methods for Studying Digital Library Use," April 2004, *Information Research*, 9 no. 3 (2004) <<http://informationr.net/ir/9-3/paper177.html>> (13 August, 2004).
- [9] Jakob Nielsen, *Usability Engineering*, (San Diego: Morgan Kaufman, 1993).
- [10] Steve Krug, *Don't Make Me Think: A Common Sense Approach to Web Usability*, (Indianapolis: New Riders Publishing, 2000).
- [11] John D. Musa, "More Reliable Software Faster and Cheaper: An Overview of Software Reliability Engineering," 7 February, 2003, <http://members.aol.com/JohnDMusa/ARTweb.htm> (13 August, 2004).
- [12] Nachiappan Nagappan, Mark Sherriff and Laurie Williams, "On the Feasibility of Using Operational Profiles to Determine Software Reliability in Extreme Programming", *Technical Report*, <<http://www4.ncsu.edu/~mssherr/papers/TR-2003-15.pdf>> (13 August, 2004).
- [13] Section 508, "The Road to Accessibility," <<http://section508.gov/index.cfm>> (13 August, 2004).
- [14] Mary Beth Rosson and John M. Carroll, *Usability Engineering, Scenario-based Development of Human-Computer Interaction*. (San Francisco: Morgan Kaufman, 2002).
- [15] Alistair Cockburn, *Writing Effective Use Cases*, (Boston: Addison-Wesley, 2001).
- [16] John M. Carroll, *Making Use: Scenario-Based Design of Human-Computer Interaction*, (Cambridge: MIT Press, 2000). JoAnn T. Hackos and Janice C. Redish, *User and Task Analysis for Interface Design*, (New York: John Wiley & Sons, 1998). Larry L. Constantine and Lucy A.D. Lockwood, *Software for Use: A Practical Guide to the Models and Methods of Usage-Centered Design*, (New York: ACM Press, 1999).
- [17] Lisa Crispin and Tip House, *Testing Extreme Programming*, (Boston: Addison-Wesley, 2003).
- [18] Cem Kaner, "The Power of 'What If...!' and Nine Ways to Fuel your Imagination: Cem

Kaner on Scenario Testing," *Software Testing and Quality Engineering Magazine*, 5 no. 5 (2003), <http://www.kaner.com/pdfs/ScenarioSTQE.pdf> (13 August, 2004).