Testing a Website: Best Practices

by

Glenn A. Stout

Senior Functional Specialist
The Revere Group
www.reveregroup.com
gstout@reveregroup.com

August, 2001

Table of Contents

Overview 3
Research Notes 3
Web Testing Challenges 3
Web Testing Best Practices 5
  Specific Testing Areas 5
    Static Testing 5
    Test Browsing 6
      Browsing the Site 6
    Functional Testing 6
      Browser-Page Tests 6
      Transaction Testing 6
    Non-Functional Testing 7
      Configuration Testing 7
      Usability 7
      Performance 7
      Scalability 8
      Security 8
  Best Processes 9
      Project Management & Planning 9
      Testing Environments 11
      Automated Testing 11
Brief Case Study 12
    Navigation Requirements and Testing 12
Summary 12
Reference List 13

Overview

This paper will discuss best practices for testing a web application, and how the System Development Lifecycle (SDLC) benefits from using these best practices.

The author of this research paper is a senior functional specialist at a consulting firm, located in Deerfield, IL. During the year 2000, the consulting firm was engaged at ABC Company, with the task of establishing a SDLC, where the author served as requirements, testing, and deployment manager.

A key process that was put in place was the testing methodology, which had an area of concentration on the process to best test an e-commerce web application.

This paper will discuss best practices for testing a web application. It will contain references to various established processes, and offer a brief case study that lends focus to certain areas discussed in the academic portion of this paper. The paper is part academic study and part case study.

Research Notes

While preparing to write this research paper, the author performed a literature search on testing best practices, and various topics for testing. While the amount of information that is available about general testing practices from peer-reviewed journals is vast, there is not nearly the magnitude of information about testing specifically for the web. However, to fill this gap, many websites have articles and best practices, written by certified quality professionals and other testers from the trenches of testing web applications. A good portion of the research material that is in this paper comes from articles from these types of websites. Although there are some references to articles from more well known sources, they are more high-level and do not delve into the details that the other types of articles discuss.

Two articles must be mentioned in this section, as they are the very best articles on web testing that the author has reviewed, and are referenced often within this research paper. They are Part 1 and Part 2 of Paul Gerrard's "Risk-Based E-Business Testing" documents. These two documents encompass all facets of web testing, and in great detail.

Web Testing Challenges

Software testing has been with us for many years, and many standards and best practices have been written about software testing for as long. The World Wide Web (WWW) has been in existence for less than ten years, and the situation where serious business critical applications are online is even more recent. These sites conduct what is known as electronic commerce (e-commerce), which is business over the WWW.

Applications that are e-commerce based, or even if they are static websites, are different, and yet similar, to traditional software applications. There are different challenges, both technical and non-technical, that apply to testing a web application. At a recent Practical Software Quality Techniques (PSQT) conference, Robert Glass noted that an underlying theme of the conference was that, due to the WWW and e-commerce applications, the testing community has moved back about two decades in testing approaches, and the traditional approaches may never be used for future testing (2000).

Testers are still learning how to best test e-commerce applications, as most of these applications are business critical, and it is still a massive and growing marketplace. Millions of dollars have been spent on websites, and the investors expect success. Unfortunately, e-commerce history is filled with expensive failures. Some of which could have been avoided by better testing before the site was opened to the general public (Samaroo, Allott & Hambling, 1999).

The repercussions of having a poorly operating website are staggering, and even affect the brick and mortar stores that the websites are enabling online. A recent study showed that when errors are found on an e-commerce website, 28% of the people stopped shopping at the site, 23% stopped buying from the site, and 6% of the people were so upset, that they stopped buying at brick and mortar store that the site is based on (Gerrard, 2000a). One can only surmise that the customers feel that if the company cannot provide a quality website, then they may not be able to sell a quality product from their stores.

Websites offer new challenges to developers, as well as testers. Scalability and performance are two areas that are significant in the e-commerce and web applications space. However, when new technology is used to make web applications perform well and scalable, new testing methodologies have to be created along with those technologies. Performance is critical, and based on a study from the Newport Group, more than half the recently deployed transaction-based web applications did not meet expectations for how many simultaneous users their applications could handle (Shea, 2000).

The nature of websites as a whole offer challenges to the testers to effectively test the sites. They tend to be built in a requirements-free, rapid-development environment, all done with limited management support (Glass, 2000). This combination of factors makes it very difficult for the testers to fully test the system to a high degree of quality.

Time is also a factor, as the concept of “internet-time” which is generally considered to be about four times faster than normal time, has served to cut the test time for web applications, which affects quality and completeness of tests as well (Hayes, 1996).

Other factors that are very important regarding web applications are security, reliability and recoverability. Uptime requirements for web applications are far more stringent than for off-the-shelf/shrink-wrap software. People expect that websites are secure, and are available twenty-four hours a day, seven days per week. When they are not, the business suffers (MacIntosh & Strigel, 2000).

A successful web application can be summarized to as being: usable, secure, scaleable, reliable, maintainable and highly available (Samaroo et al, 1999). If a site does not meet that criteria, then it may run into problems, so all of these factors need to be tested, and that is a complex process.

To summarize the challenges of web testing, it can be said that the normal testing challenges exist, such as requirements quality, etc. but the impact is much higher, due to the critical nature of websites, e-commerce websites in particular. Additionally, there are other non-traditional testing efforts that are required for websites, such as performance testing, scalability, and others which add further challenges to a website testing team.

Web Testing Best Practices

In light of the challenges described in the prior section, there are solutions and best practices that many in the testing field have published in various forums. While performing a literature search for best practices regarding website testing, many documents that were found were papers, similar to this research paper, describing best practices for website testing. By taking the most critical or best practices from these many sources, it is the author's goal to provide a collection of the very best of these best practices. This is not necessarily a complete end-to-end methodology, but a collection of additions that can be made to a methodology that may already be in place.

The best practices will be divided into two main sections, "Specific Testing Areas" and "Best Processes." The first section will discuss specific types of tests and best practices, where the second will discuss specific processes for certain areas of testing.

Specific Testing Areas

These are specific testing areas that require special attention regarding website testing. There are many areas of web testing that are presented in this research paper, and equally many not presented. The author, based on his experience, chose the tests that are presented as either the most important types of tests, or the ones that are not executed as much as they should be. Each section will be briefly explained, and then best practices will be discussed.

Static Testing

Static testing is the testing of the objects in a web browser that do not change, or are not transaction based. This type of testing is done on a web page that has already been loaded into a web browser. There are several types of static testing, and they will be discussed in this section.

Content Checking

Once the web page has been loaded, it has to be tested for accuracy, completeness, consistency, spelling and accessibility. These terms have the traditional meanings, and the tests are as elementary as they sound. However, it is in areas like these where the site is first judged by the website visitor. For example, if the site has numerous misspellings, the product that the website is offering may come into question as the visitor may feel that if the attention to detail is not given to the site, it may not be given to the product either. These tests are mentioned in this research paper as these are simple things that may not automatically be on a web tester's test plan, as most of these are unique to the web (Gerrard, 2000b).

Browser Syntax Compatibility

This test is one level below the actual content. It is the technology of how to represent the content, whether that content consists of text, graphics, or other web objects. This is an important test as it determines whether or not the page under test works in various browsers. Even regarding only Microsoft's Internet Explorer and Netscape's Navigator web browsers, this is a significant issue due to the fact that there are many versions of both still in use. These versions do not work the same, and depending on what the minimum version and browser type requirements are, the pages need to be tested in each supported browser (Gerrard, 2000b).

Visual Browser Validation

Simply, does the content look the same, regardless of supported browser used? If the requirements are that only one browser and version will be supported by the application, this test

is not necessary. However, even if more than one version of the same browser will be supported, the page under test should loaded into both browsers, and they should be visually checked to see if there are any differences in the physical appearance of the objects in the page. If there are, they may be things such as the centering of objects, table layouts, etc. The differences should be reviewed by the users to see if there is any need to change the page so that it appears exactly the same (if possible) in all of the supported browsers (Gerrard, 2000b).

Test Browsing

Test browsing tests aim to find the defects regarding navigation through web pages, including the availability of linked pages, and other objects, as well as the download speed of the individual page under test. The integration of web pages to server-based components is tested, to ensure that the correct components are called from the correct pages (Gerrard, 2000b).

Browsing the Site

When traversing links and opening new pages, when a new page is opened, several questions should be addressed on each and every page the system links to. Can the page be downloaded and displayed? Do all objects load in an acceptable time ("acceptable" would be based on the business requirements)? When user turns the browser option of "images-load" to "off" – does the page still work? Do all of the text and graphical links work? All of these questions are important, as if the answer to any of them is in the negative, it would be considered a defect (Gerrard, 2000b; Smith, 2001).

Other issues to validate are whether the site still works if JavaScript or Java is disabled, or if a certain plug-ins is not loaded or disabled. A good test case is to use a browser with no plug-ins loaded during testing, and when the tester is queried to download a plug-in, they should not load them, and see how the site reacts without the plug-in (Smith, 2001).

Functional Testing

Browser-Page Tests

This type of test covers the objects and code that executes within the browser, but does not execute the server-based components. For example, JavaScript and VBScript code within HTML that does rollovers, and other special effects. This type of test also includes field validations that are done at the HTML level. Additionally, browser-page tests include Java applets that implement screen functionality or graphical output (Gerrard, 2000b).

Transaction Testing

This is a critical test in an e-business application. This type of test is designed to force the software to invoke the various components as a complete set and to determine whether the direct and indirect interfaces work correctly. These interfaces are: Transfer of control between components, transfer of data between components (both directions) and consistency of us of data across components (Gerrard, 2000b). At a higher level, a test case within this set would confirm that information entered by the user at the web page level makes it to the database, in the proper way, and that when database calls are made from the web page, the proper data is returned to the user.

Non-Functional Testing

Configuration Testing

Beyond the browser validation, this type of test takes into consideration the operating system platforms used, the type of network connection, internet service provider type, and browser used (including version). The real work for this type of test is ensuring that the requirements and assumptions are understood by the development team, and that a test environment with those choices are put in place to properly test it (Gerrard, 2000b).

Usability

For usability, the tests can be subjective, but there are standards and guidelines that have been established throughout the industry, and it would be easy for a project team to blindly follow them, and feel that the site will be acceptable since the standards are followed.

However, human-computer interaction standards and guidelines cannot guarantee a usable website. Designers should not rely on them for all or even most of the design decisions, and project managers should not let standards compliance lull the team into complacency, thinking that since the standards are followed, that the site will automatically meets the needs of the users, their tasks, and their work environment (Buie, 1999).

A proactive suggestion is that while establishing the design guidelines, to define requirements that can be positively identified and measured (Gerrard, 2000b). A way to do this is to capture and quantify the meaning of learnability, understandability, and operability in a testable form. These concepts need to be formulated into testable requirements by describing how these concepts will be accomplished (MacIntosh & Strigel, 2000).

Performance

Performance testing is the validation that the system meets performance requirements. This can be as simplistic as ensuring that a web page loads in less than eight seconds, or can be as complex as requiring the system to handle 10,000 transactions per minute, while still being able to load a web page within eight seconds. The section below offers best practices for the execution of performance testing.

During research, one topic that was repeated was the importance that the performance-testing server is exactly like the production server; in fact, it ideally should be an exact replica in every way. It is very important to make sure every component (networks, firewalls, servers, mainframes) matches the production equipment. The weakest link theory applies here. For example, even the best server cannot make a difference if the firewall machine, and more importantly the number of firewall rules, are not the same (Hagen, 2000).

In the execution of performance tests, Weyuker and Vokolos (2000), who provide a detailed case study on performance testing, outline typical steps to create performance test cases. These are:

- Identify the software processes that directly influence the overall performance of the system.
- For each of the identified processes, identify only the essential input parameters that influence system performance.

- Create usage scenarios by determining realistic values for the parameters based on past use. Include both average and heavy workload scenarios. Determine the window of observation at this time.
- If there is no historical data to base the parameter values on, use estimates based on requirements, an earlier version, or similar systems.
- If there is a parameter where the estimated values form a range, select values that are likely to reveal useful information about the performance of the system. Each value should be made into a separate test case.

Performance testing can be done through the "window" of the browser, or directly on the server. If done on the server, some of the performance time that the browser takes is not accounted for. Scripting GUI orientated transactions to drive the browser interface can be much more complicated and the synchronization between test tool and browser it not always reliable. Therefore, if testers decide to ignore the performance time taken by the browsers, it is important to get "buy in" from project team members and users to understand the compromise. If there are issues, testers should advise management that performance-testing using the GUI will introduce a time-intensive effort that may or may not impact the project timeline (Gerrard, 2000b).

To assist with load and performance testing, testers should use the test scripts that have been created early in the project as a basis for initial load testing. By using the existing scripts, this avoids rework and allows the scripts to be used at different times by different virtual users when validating system performance (Courtney, 1999).

Scalability

The term "scalability" can be defined as a web application's ability to sustain its required number of simultaneous users and/or transactions, while maintaining adequate response times to its end users (Shea, 2000).

When testing scalability, configuration of the server under test is critical. All logging levels, server timeouts, etc. need to be configured just like production. In an ideal situation, all of the configuration files should be simply copied from test environment to the production environment, with only minor changes to the global variables (Hagen, 2000).

In order to test scalability, the web traffic loads must be determined to know what the threshold requirement for scalability should be. To do this, use existing traffic levels if there is an existing website, or choose a representative algorithm (exponential, constant, Poisson) to simulate how the user "load" enters the system (Hagen, 2000).

Security

Security is a critical part of an e-commerce website. Best practices to best test how secure the site is are in this section.

The security best practices listed below come from an outstanding article on website security testing by Russ Smith (2001).

There are several areas of security, and below them are questions or issues that should be answered for each section.

Data Collection: Web sites collect data in log files, as well as through forms in which users supply to the website information that is saved on the web server.

- The web server should be setup so that users cannot browse directories and obtain file names. For example, www.example.com/data should not list the files in that folder.
- Data should be secured internally so unauthorized employees do not have access.

Get vs. Post: Technically, there are two ways that information can be collected from a web page when a user submits a web form, one is a GET and one is a POST. A GET shows some information in the Uniform Resource Locator (URL) that could be sensitive, where the POST does not.

- Early in the project, encourage developers to use the POST command wherever possible.
- When testing, check URLs to ensure that there are no "information leaks" due to sensitive information being placed in the URL while using a GET command.

Cookies: A cookie is a text file that is placed on a website visitor's system that identifies the user's "identity." The cookie is retrieved when the user re-visits the site at a later time. Cookies can expire in a short period of time, such as minutes or hours (session cookie) or can last for months or years (persistent cookie).

- Cookies can be controlled by the user, regarding whether they want to allow them or not. If the user does not accept cookies, will the site still work?
- Are the cookies necessary? Cookies should be used judiciously so that if users have their browsers at the setting to "provide warning message when cookies are used" and multiple cookies are used, the user will be inundated with multiple warning boxes throughout their visit, perhaps discouraging them from re-visiting the site.
- Is sensitive information in the cookie? If multiple people use a workstation, the second person may be able to read the sensitive information saved from the first person's visit. Information in a cookie should be encoded or encrypted.

Best Processes

In this section, the "best practices" for testing processes that specifically pertain to web testing are discussed. They fall into certain categories, and relate to the actions that orbit the actual test, such as project management issues regarding web testing, and ways to implement testing tools.

Project Management & Planning

This section will deliver examples of best ways to manage a website testing project, and how to best plan the various tests that are needed for web testing.

For planning testing, it is clear from Paul Gerrard that waiting until after the executable software has been built is the most costly and least effective way of testing. He, and other sources concur, suggests that the testing be done throughout the entire lifecycle, in parallel with every stage of development, as is the least costly and most effective way to test (2000a).

Also, by testing as early as possible in the development cycle, the faults are detected and corrected in the development stage, where they are less expensive to correct (Samaroo, Allott & Hambling, 1999).

To follow on with the idea that testing needs to be a part of every phase of the project, testability needs to be an explicit goal of the design and implementation. Successful testing begins in the

product requirements phase of the project (Ritter, 1999). In particular, performance testing must be an integral part of designing, building, and maintaining the web applications. A validation of the system architecture in the early stages of development is as critical as the eventual performance test of the working system. (Shea, 2000)

Another thing that the designers and developers can assist with regarding testing is to engage the test server's ability to provide application log monitoring. To take advantage of this, the developers have to instrument the code with special calls to an application log. This may take extra time, but it is worth it in the long run. The log will show all of the database calls that the system makes. The round trip time for the database request can also be captured, which assists with performance testing (Hagen, 2000).

When projects get behind, or even to get ahead of schedule, a project manager should consider using non-testers to do the simple regression tests that the regular tester are very familiar with. Often these people find defects that normal testers may miss. This frees up time for the testers to perform the more complex tests (Wilson, 2001).

Since user interface is generally the last part of the application to be written, server-based components cannot be tested fully until late in the project. This is important to keep in mind when planning the testing schedule. Although a first pass at a server-based component worked early on the project using a special test user interface, until the final one is completed and re-tested, this test cannot be considered complete (Gerrard, 2000b).

Requirements management is critical, and project managers need to control changes to requirements and scope, as serious testing is impossible in an environment of uncontrolled changes to requirements (MacIntosh & Strigel, 2000).

When planning testing, clear testing objectives and criteria for successful completion need to be set, which includes test coverage measures. The test program must be properly planned, with test scripts giving precise instructions and expected results (Samaroo, Allott & Hambling, 1999).

Performance and Scalability Testing

As discussed earlier in this research paper, performance and scalability testing are very important facets of web testing. Some techniques to best test those areas are in this section.

Most importantly for performance testing, it is critical to have the performance requirements explicitly stated in a concrete, verifiable manner. This can be stated in terms of throughput or response time for a set number of users. The problem of not getting these requirements can sometimes be solved by making it an issue when the architecture is scheduled, and enforce a decision at that time, by not permitting the project to proceed until performance requirements are determined (Weyuker & Vokolos 2000).

To plan for performance testing, having server monitoring enabled is important, especially on the servers responsible for creating the web pages (Hagen, 2000). Additionally, planning to have the right amount of data in the test database is important. Testers must ensure that both the test machine and production have equal or at least very similar levels of data. This will enable the testers to see how the application will perform under realistic conditions (MacIntosh & Strigel, 2000).

Things that can affect performance must be kept in mind when designing the system. For example, database-indexing techniques can have a significant impact on site performance when done incorrectly. This is especially negative when database requests are either unable to find information or get trapped inside an infinite loop while trying to fill a request (Shea, 2000).

The security requirements also impact performance. What must be decided is to what degree security is necessary. Both the necessary and unnecessary layers of encrypting and decrypting transaction data impact response time. Based on the requirements, the layers should be minimized where possible (Shea, 2000).

Testing Environments

Using a proper testing environment is always a significant testing concern, but it is critical regarding web testing. Some best processes for test environments are in this section.

Many of articles reviewed mentioned that having a separate test environment was critical. For the entire project, a minimum of three environments is suggested. The first is a development server for the development team, the second a test server that is periodically updated with the new releases, and the third is the production server. If the test server crashes, neither development nor production environments are affected (MacIntosh & Strigel, 2000).

It is important that the test environment be effective, and exactly the same as the production environment (Samaroo, Allott & Hambling, 1999). Ensure that it is completely independent of the production environment (Crispin, 2001). It is important to obtain a commitment to a test system as early as possible in the project planning stage so that it can be treated as a project requirement, and not just a luxury system that can be dropped if the schedule slips (Driscoll, 1997).

Automated Testing

The research that was done shows that automated testing is a large and important part of website testing. In this section, process suggestions are noted.

Automated testing is critical to a quality website, as the chances of getting adequate testing done in the tight time scales for an e-commerce project and without automation are extremely slim (Samaroo, Allott & Hambling, 1999; Courtney, 1999).

Regarding the creation of automated test scripts, consider working with the development team to help them automate their tests, and get help automating the test team's scripts. If the same tool is used, some developer scripts can be transitioned to the testing team, to be used as a regression test set (Gerrard, 2000a).

To automate the checking for links, it is suggested that a using a link checker, which can verify links through a file system instead of the web, is an optimal solution. This tool could speed up the verification process considerably, possibly up to 10 times faster than through a server (Driscoll, 1997).

Another requirement for an automated tool is that it should be able to check the links to linked pages, frame pages, images used for navigational buttons, form handlers, applets, and other files (Gerrard, 2000b).

When choosing an automated test tool, it is suggested to use a Hypertext Transfer Protocol (HTTP) testing tool as opposed to a Graphical User Interface (GUI) testing tool. The advantage is that the entire HTTP response can be captured for future comparisons. Further, these test results can be viewed later on different browsers so that the test team can see how a given test would have looked had it run on a particular browser (Driscoll, 1997; Ritter, 1999).

In general, testers should automate where appropriate. There are advantages and disadvantages of automated testing. When not used, some projects suffocate under an avalanche of manual test scripts. When employed, testing tools management and the creation of automated testing scripts can consume all available time and resource. As an overall guideline, testers should automate regression tests to the furthest extent possible as a minimum set of automated scripts (Ritter, 1999).

Brief Case Study

The author was testing manager for a website application, and although the application was rolled out successfully, there were still some lessons learned regarding the testing effort.

Navigation Requirements and Testing

Gathering requirements, and testing to those requirements, is a key process flow that needs to take place in a project. It was advantageous that the author was both the requirements manager and the testing manager for the project, which allowed for the flow of requirements to the testing team in an efficient manner. However, the requirements that were not gathered, or missed, and were not tested, were navigation-type requirements.

These types of requirements are based on what happens when the user goes from one screen to the next, and possibly back again. A key issue was what would happen to data that was entered on a form, once the user went to the next form, and then back again. Would the data still be there? Would pull-down data that was selected still be what the user chose? These were critical issues that the users demanded be fixed when discovered during user acceptance test.

Had these requirements been determined early in the project, it would have been far less expensive to provide the functionality when done during the development process, instead of being identified as defects, and being fixed, during the testing phase.

Summary

It is clear that website testing is a new breed of testing, and will continue to change for years to come. In this research paper, the two main sections, Specific Testing Areas and Best Processes discussed ways to test websites and process enhancements to current SDLC processes respectively. The brief case study described some lessons learned regarding web testing, and ways to avoid the same issues when engaged in a website development project. Web testing is a challenging exercise, and by following the best practices and processes described in this research paper, some of those challenges may be mitigated.

Reference List

Buie, E. (1999). HCI Standards: A Mixed Blessing. *Interactions, 6* (2), 36-42.

Courtney, P. (1999, July). Testing e-commerce. *Applications Development Trends,* 24-34.

Crispin, L. (2001). *Stranger in a Strange Land: Bringing QA to a Web Startup*. Retrieved June 15, 2001, from the World Wide Web: http://www.stickyminds.com/docs_index/XUS247559file1.doc

Driscoll, S. (1997). *Systematic Testing of WWW Applications*. Retrieved June 15, 2001, from the World Wide Web: http://www.oclc.org/webart/paper2/

Gerrard, P. (2000a). *Risk-Based E-Business Testing: Part 1 – Risks and Test Strategy*. Retrieved June 15, 2001, from the World Wide Web: http://www.evolutif.co.uk/articles/EBTestingPart1.pdf

Gerrard, P. (2000b). *Risk-Based E-Business Testing: Part 2 – Test Techniques and Tools.* Retrieved June 15, 2001, from the World Wide Web: http://www.evolutif.co.uk/articles/EBTestingPart2.pdf

Glass, R. (2000). *Has Web Development Changed the Meaning of Testing?* Retrieved June 15, 2001, from the World Wide Web: http://www.stickyminds.com/swtest.asp?zone=WBTST&sid=331706&sqry=%2AJ%28ARTCOL%29%2AR%28createdate%29%2AK%28topicarea%29%2AA%28WBTST%29%2A&sidx=13&sopp=10&ObjectId=2163&Function=DETAILBROWSE&ObjectType=COL

Hagen, M. (2000). *Performance Testing E-Commerce Web Systems Presentation Paper*. Retrieved June 15, 2001, from the World Wide Web: http://www.stickyminds.com/docs_index/XDD2070filelistfilename1.doc

Hayes, L. (1996, July). Testing distributed applications: Unraveling the Web. *Datamation*, 108-114.

MacIntosh, A. & Strigel, W. (2000). *"The Living Creature" – Testing Web Applications*. Retrieved June 15, 2001, from the World Wide Web: http://www.stickyminds.com/docs_index/XUS11472file1.PDF

Ritter, D. (1999, February). Web Testing Is Not an Oxymoron. *Intelligent Enterprise*, 66-69.

Samaroo, A., Allott, S., & Hambling, B. (1999). *E-ffective Testing for E-Commerce*. Retrieved June 15, 2001, from the World Wide Web: http://www.stickyminds.com/docs_index/XML0471.doc

Shea, B. (2000, May/June). Avoiding Scalability Shock. *Software Testing & Quality Engineering Magazine*, 42-46.

Smith, R. (2001, Jan/Feb). Behind Closed Doors: What every tester should know about web privacy. *Software Testing & Quality Engineering Magazine*, 43-48.

Weyuker, E & Vokolos, F. (2000). Experience with Performance Testing of Software Systems: Issues, an Approach, and Case Study. *IEEE Transactions on Software Engineering*, *25* (12), 1147-1156.

Wilson, J. (2001). *Testing in Internet Time.* Retrieved June 15, 2001, from the World Wide Web: http://www.stickyminds.com/docs_index/XDD2540filelistfilename1.doc