Web Performance Measurements: Full Browser vs. Website Tester

By Peter Sevcik
October 2008

If you need to assess the performance of your enterprise's website, there are some things you should know about differing approaches to measuring performance using synthetic testing services. There are two testing approaches: one that uses a pared down subset of browser functionality packaged into what we call a 'website tester', and another that uses a fully functioning browser. NetForecast performed research to assess which approach gives the most accurate website performance measurement results.

NetForecast found that the full browser-based testing alternative delivers the best results, especially for media-rich sites, because it more accurately measures the true end user experience by executing all of the software and loading all of the content. Traditional website tester-based services in contrast, were designed to measure performance in a world dominated by Flat HTML, and therefore do not measure the performance of a complete range of website elements.

This report describes the NetForecast research, presents the research results, and makes recommendations based on our findings.

The NetForecast Research

There are three measurement elements in a website experience. The first is the time it takes the server to "flip" requested content to the network. Second is the time it takes for the content to traverse the Internet. And the last element is the time it takes to render the content on the user's machine. Server response time is usually extremely fast because most popular content is delivered from the server's cache. Network time is understood to be highly variable due to geographic distances, users' access bandwidth, and frequent network congestion events. Thus the network component should be well measured and tracked.

The time it takes to render varies considerably depending on the particulars of a user's machine. In the early days of the web the high render time variability was due to desktop configuration and browser differences. Today the variability is exacerbated by the complexity and demands of rich media applications that employ new client delivery tools such as JavaScript, Flash, Flex, and Ajax.

This report focuses on having a comprehensive measurement of the all three response time components. Focusing on just one has limited value since you may be missing the component which is the most significant contribution to response time as seen by the user.

NetForecast gathered total response time information for three popular home pages—MSNBC, AOL, and Microsoft—using two synthetic measurement services, one browser-based, and the other website tester-based. Measurement locations were distributed across the US in California, Florida, Minnesota, New York, and Texas, and testing locations for both services were paired to be within the same state to normalize for geography.

The browser-based testing service used a test script operating within the browser. The script consisted of simple URLs for each of the three tested home pages. The service

NetForecast Report 5094

downloaded and rendered all of the content on the home pages including that called for by JavaScript, and the service measured the time taken to complete the entire operation.

The website tester-based service used a tester operating a subset of browser functionality. The test script consisted of the same URLs as the browser-based script, and the service downloaded all of the content marked as "gets" within the HTML code of the base homepage. The service did not execute nor load content asked for by JavaScript, XML, or AJAX software.

Test Results

NetForecast found significant and consistent differences in the average response time results for the browser-based service compared with the website tester-based service. Because the browser-based testing service approach more accurately reflects the actual end user experience, average response times were longer than for the website tester-based service. This is because a more realistic browser experience includes more application turns (application client-server software interactions), larger payloads (bytes delivered), and more desktop execution time (rendering) than the website tester alternative.

The significant and consistent difference in results from the two approaches is clearly seen in the average response times for the MSNBC home page shown in Figure 1. The average browser-based testing response time is consistently about six seconds slower than for the website tester-based service throughout the entire testing period.

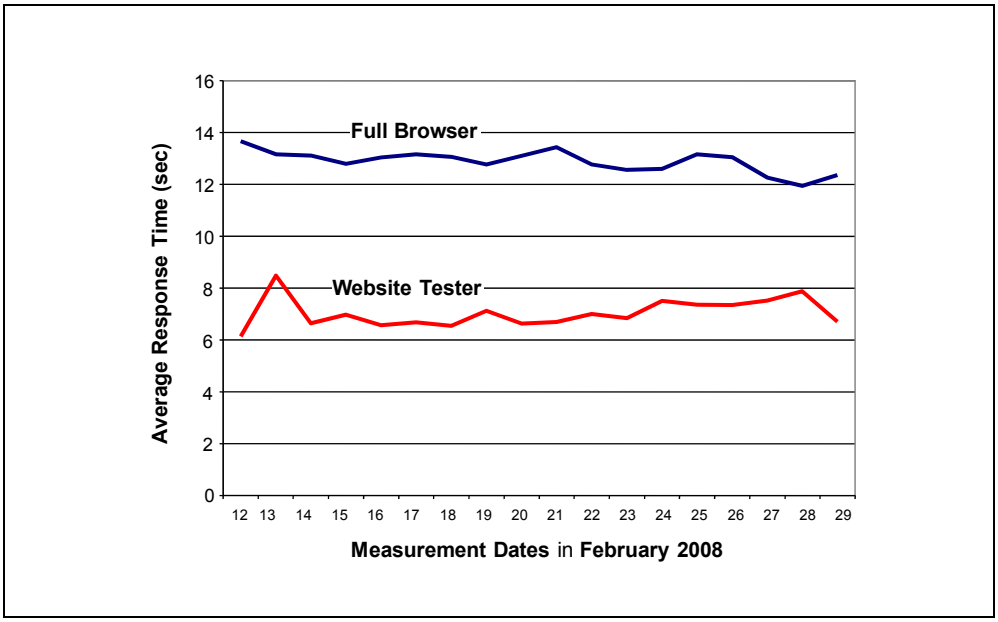

Figure 1 – MSNBC Home Page Results

Comparative test results for the Microsoft home page in Figure 2 are informative because at the start of the test period the browser-based test response times are approximately double those of the website tester response times. Then, about two thirds of the way through the test period, Microsoft apparently made a change to the website that dramatically improved its performance. That change is readily apparent in the browser-based results, but it is not reflected at all in the website tester-based. This shows how much additional information can be gleaned from browser-based testing.

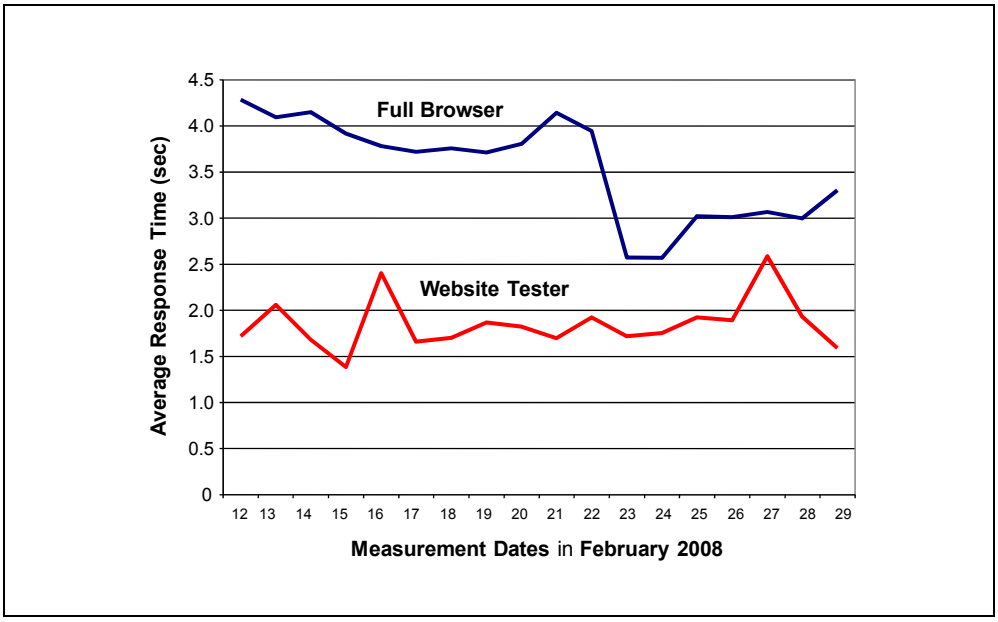

Figure 2 – Microsoft Home Page Results

Measurement Consistency Hint

A word to the wise gleaned from our testing. It is important that each measurement be consistent such that differences are true indications of performance changes and not test artifacts.

Ensure that the browser caches are flushed between browser-based tests. We noticed some anomalous results that were explained by the fact that the browser caches were not flushed between tests. If you see results that look too good to be true, check your cache settings in the test scripts or measurement set-up.

Test results for the AOL home page in Figure 3 also show how browser-based measurements can be more informative than website tester-based measurements. For most of the test period, the browser-based results are one to two seconds slower than the website tester-based results. About three quarters of the way through the period, a change degraded the user experience. Although this is clearly reflected in the browser-based results, if the website tester-based results were all you had to work with, you would not see the change at all.

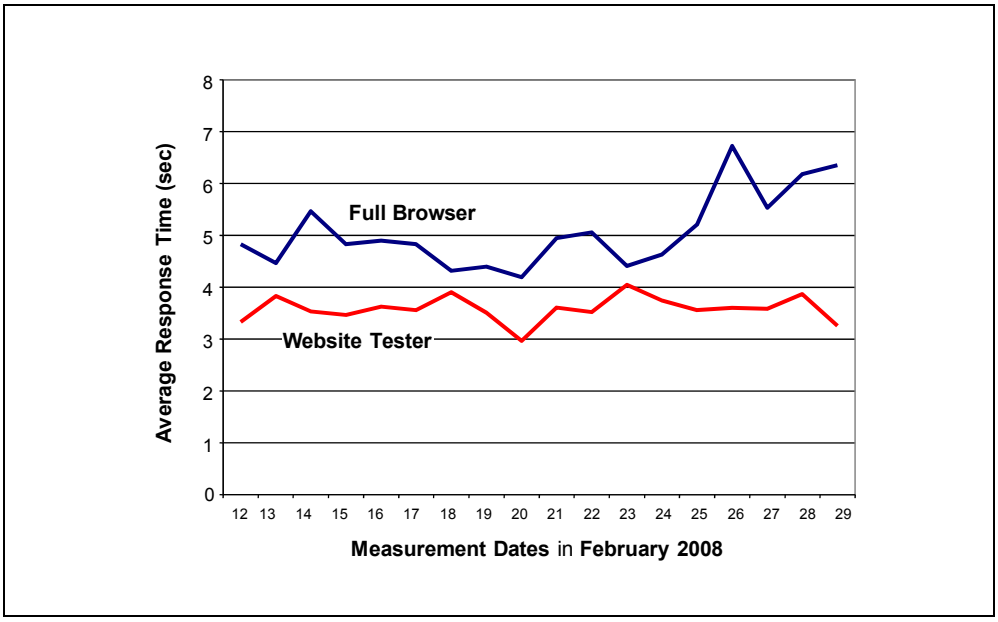

Figure 3 – AOL Home Page Results

Conclusions

Web applications and websites are becoming increasingly media rich. No longer HTML-centric, many websites now routinely contain JavaScript code, XML, AJAX, Flash, Flex, and other sophisticated content. To accurately reflect the actual end user experience, synthetic test approaches need to exercise all of these capabilities.

Traditional performance measurement approaches test only a limited part of a website and thus miss much of the true end user experience time. These approaches deliver metrics such as time to first byte, time to load the base page, and time to load the content referenced on the base page. In our experience website tester-based measurement services not only fail to test all aspects of the site, they do not retrieve information as a real browser would. For example, rather than fetch page elements in the sequence that the browser would follow, they may fetch them simultaneously. As informative as the resulting metrics may be, they constitute only a part of the true load time for a media-rich page.

In short, capturing "near real world" metrics requires a test approach that uses a fully functioning browser executing all of the software and loading all of the content as directed by the website.

Recommendations

Browser-based synthetic testing services are generally more costly than their website tester-based counterparts. If your website is composed primarily of HTML, or if you require only rudimentary information about your website's performance, it may suffice to use the less expensive, less sophisticated alternative.

But if your site is or is becoming media rich, your measurement data should more accurately reflect your end user's actually experience. In this case we recommend using a browser-based synthetic testing service.

Peter Sevcik is a principal of NetForecast and the founder of the Apdex Alliance. He is a leading authority on measuring, assessing, and improving the performance of networked applications. Peter has contributed to the design of more than 100 networks, including the Internet, and to the success of more than 25 application management products. He can be reached at peter@netforecast.com.

NetForecast helps enterprises understand and improve the performance of networked applications.

Additional information is available at: www.netforecast.com