Software

Four Ways to Modernize Your Application Performance Monitoring Strategy for Web 2.0 and AJAX

Abstract

As businesses become increasingly reliant on online channels for customer interactions, the need for faster, more functional websites becomes increasingly evident. While many Web 2.0 approaches aim to increase the speed and efficiency of web applications, they also present a challenge to traditional application performance monitoring (APM) solutions. With much of the processing taking place within the browser itself, or through calls to third party services, data center-based monitoring solutions find themselves faced with significant blind spots.

Because this activity frequently does not generate calls back to the organization's network, it becomes necessary to also monitor activity at the edge, from within the browser itself, to obtain a complete picture of application performance. This paper prescribes four ways in which today's monitoring technology must adapt to adequately meet the challenges presented by Web 2.0, and how Foglight, Dell's APM solution, addresses these challenges.

Increasing the speed and efficiency of web applications

The website has become the primary way a company interacts with its customers—if not the only way. Consider, for example, the explosive growth of the online channel for e-commerce. According to a March 2013 report[1] by Forrester Research, Inc., online retail sales in the US will reach $370 billion by 2017—a 10 percent compound annual growth rate (CAGR) over the next five years. To stay competitive, brick-and-mortar retailers have had to shift their focus to the online channel to avoid becoming a showroom for Amazon.com and other e-commerce websites. The trend is not just limited to retailers. Websites provide a lower cost option for delivering both products and services, that's far more cost-effective than live support at a physical location.

[1] *US Online Retail Forecast, 2012 to 2017, by Forrester Research, Inc.*

Online retail sales in the US will reach $370 billion by 2017 – a 10 percent compound annual growth rate (CAGR) over the next five years.[1]

The consumerization of IT is also contributing to the increasing importance of the online medium. Consumers are demanding access to more services, from more devices, from any location. According to Flurry Analytics, smart device adoption is 10X faster than that of the 1980s PC revolution, 2X faster than that of the 1990s Internet boom and 3X faster than that of recent social network adoption.

As a result, the online experience – and the need to make websites faster and more functional – has become paramount. Several trends have thus emerged in web application design, often enabled by Web 2.0 technologies, such as JavaScript and AJAX, including:

- **Fewer page loads.** Many e-commerce sites simplify the buy process by reducing the number of pages a user has to move through before checkout. For example, a consumer configuring a computer for purchase is able to edit the configuration without having to reload the entire page for each change he or she makes.
- **Asynchronous page loading.** Typically employed for pages that are mostly HTML, this approach accelerates performance by first synchronously loading the HTML code that loads quickly, then asynchronously loading slower elements. For example, a home page is quickly loaded synchronously, fully displayed and useable. Then banner ads and content areas below the fold (that are not visible initially) are loaded and displayed asynchronously.
- **Client-only processing.** Rendering events on a page can now happen without any interaction with a back-end server. In this case, the content is supplied as part of the page by the web server but is not displayed unless triggered by an event – for example, when a user hovers a mouse over a dropdown menu causing a list to be displayed.
- **Content delivery networks (CDNs).** Frequently, HTTP requests from the browser are fulfilled by CDNs or other caching technologies, without having to reach back to the web server. The intent is to provide faster access to static content such as images. For example, Akamai caches entire pages on the Akamai edge.
- **Calls to third-party service providers.** In many cases, calls are made from the browser directly to third-party providers and consequently don't reach the web server. Examples include embedded social media plugins or a Google map widget for location information.
- **Single-page applications.** This is the newest form of asynchronous interaction in which the entire application consists of a single page and operates in a manner similar to a desktop application. The page is loaded, followed by numerous asynchronous calls within the page that are triggered by user actions. This approach improves performance and reduces network load with single calls to access only what is needed from the server. Gmail is a good example of this.

The Web 2.0 challenge in APM

Prior to Web 2.0, you could track user activity simply by monitoring HTTP page requests and their associated responses. The web server would respond by returning content to the user's browser in the form of a full page. The monitor needed only to associate the individual requests with the higher level page, so both aggregate and single-request performance could be monitored at the HTTP request level. Because all requests were sent back to the web server, effective monitoring could be accomplished through agents on the web server tier or by sniffing packets travelling on the wire. This was the approach taken early on by APM solutions.

With Web 2.0, browsers have the ability to execute code embedded within a web page, eliminating the need to rely on calls to back-end application servers for code operations. JavaScript is the most popular language used for this purpose, and has fast become one of the most popular programming languages on the web due to its speed, efficiency and ability to reduce network load by running locally on end-user hardware. JavaScript can be

[1] *US Online Retail Forecast, 2012 to 2017, by Forrester Research, Inc.*

used for such activities as animating page elements, playing audio and video, and validating web form input data.

The broader trend of AJAX (Asynchronous JavaScript and XML) programming additionally provides the ability for browsers to make asynchronous requests, eliminating the need to reload an entire page when portions of the page are updated. For example, when a user is on a checkout page and requests shipping charges, he or she can see the shipping cost appear without having to fully reload the page.

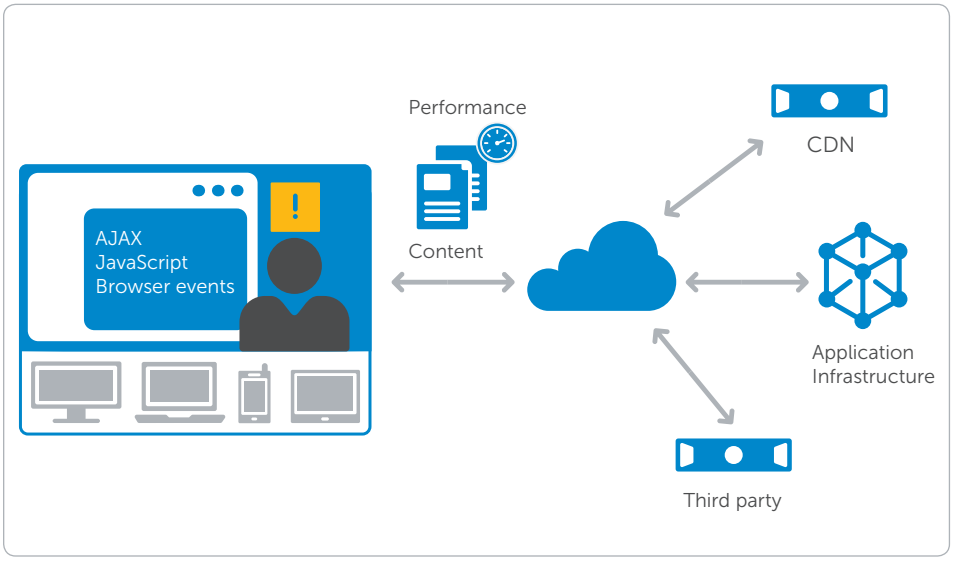
AJAX and JavaScript introduce a number of challenges when using traditional methods to monitor web applications. Some key blind spots associated with these methods include:

Inadequate code-level analysis
APM solutions traditionally focused on monitoring code execution by installing agents on data center servers. This approach now only tells part of the story. As much as 80% of the code in a modern web application may execute within the browser. Analogous to bytecode instrumentation within the application server, browser-side instrumentation is necessary for monitoring JavaScript execution and errors.

Incorrect page response times
It is no longer possible to obtain accurate page response times through monitoring of network traffic alone. Only the HTTP request and response for each individual object (or hit) that goes back to the web server (or origin point) can be timed using this approach. With Web 2.0, however, many requests may not go back to origin at all, such as those that are rerouted to CDNs or fulfilled in some other way using caching technologies. Calls to third-party web services (e.g. a Google Maps widget) cannot be accounted for through network sniffers either. To be fully inclusive of ads, maps, shopping carts, web analytics, social media modules, CDN and DNS response times, etc., page load time must be monitored from within the browser itself.

Insufficient context
At best, network traffic monitoring can associate back-end calls with the page they came from. While this would have provided sufficient context for troubleshooting a traditional application, this is no longer the case with AJAX, where there may be hundreds of

As much as 80% of the code in a modern web application may execute within the browser.



As websites become increasingly dependent on dynamic content and third party services, the true quality of end-user experience is measurable only from within the browser itself.

Supplementing your existing monitoring with exciting new instrumentation techniques can deliver more insight than ever before.

calls from a single page. Even more challenging is the fact that many JavaScript events (for example, a mouse-click on a menu item) do not create calls to web servers at all, making these browser-only events invisible to network sniffers and web server monitors.

As websites become increasingly dependent on dynamic content and third party services, the true quality of end-user experience is measurable only from within the browser itself.

Four ways to modernize your APM strategy

As with any new technology, Web 2.0 presents both challenges and opportunity. While traditional methods are no longer adequate on their own, supplementing your existing monitoring with exciting new instrumentation techniques can deliver more insight than ever before. Here are four ways to update your APM strategy for Web 2.0.

1. **Capture functional issues and establish context**
   Performance is not all that matters when it comes to external-facing applications. Application functional issues are far more common than performance issues, and a major factor in abandonment and conversion rates. Since the web application is often the only interaction point a company has with its customers, troubleshooters will not typically have the opportunity to ask the user what happened. Consider, for example, an error caused by a leading zero in a zip code field that the application was not designed to parse. Implementing a solution that captures browser events, such as mouse clicks and keyboard input data, and has the ability to replay a user's session activity, will help you proactively identify and troubleshoot these kinds of problems.
2. **Capture and troubleshoot JavaScript errors**
   Consider what would happen if a company rolled out a new AJAX feature for placing online orders that started returning JavaScript errors. There would be no indication in the web logs, and all response times would look okay. As a result, troubleshooters wouldn't become aware of the problem until customer complaints began pouring in. In this case, ignorance wouldn't be bliss, but rather lost revenue. Your APM solution should be able to detect and alert on JavaScript errors immediately for you to move quickly to resolve them.
3. **Look for detailed insight into page load times**
   To be fully inclusive of ads, maps, shopping carts, web analytics, social media modules, CDN and DNS response times, etc., page load time must be monitored from within the browser itself. Luckily, we have the HTML 5 navigation timing feature available in newer Internet Explorer, Firefox, and Chrome browsers. It includes full page load times broken out by DNS lookup, redirect, SSL handshake, processing, and cache access timing. Look for an APM solution that integrates this feature.
4. **Isolate problems to specific individual page elements**
   Currently the information captured by the browser is limited to full page loads and

	Actual end user time (incl, DNS, 3rd party, CDNs, etc)	Hit times for page objects (e.g., .css, .jpg, .gif)	API time for REST web service (cells to origin only)	Actual back-end time	Java script errors	Java or .NET execution
Network sniffer		X	X	X		
Browser instrumentation	X				X	
Server instrumentation				X		X

A multi-pronged approach is necessary for maximum visibility into modern web application performance.

does not provide timing information on individual page hits, such as loading of graphics or images, CSS stylesheets, or back-end calls to web servers or REST APIs. Network sniffers with web page analysis capability can time HTTP request and response for individual page objects, enabling troubleshooters to isolate issues related to specific page elements. Make sure your network monitoring solution includes this feature.

Monitoring Web 2.0 applications with Dell Foglight

Foglight Application Performance Monitoring from Dell delivers a collaborative customer-centric approach to APM, providing IT and the business alike with answers, not just data. When optionally combined with other Foglight capabilities for database, virtualization, or network performance monitoring, Foglight APM complements a full range of enterprise monitoring strategies.

Combining the best of traditional and modern approaches, Foglight:

- Captures every click by every user and replays web user activity for a true replication of the user experience, enabling contextual forensics and troubleshooting
- Maps your application and infrastructure dependencies with rich auto-discovery of the application runtime architecture
- Ensures accurate identification of problem areas and rapid triage with detailed response time breakdown and complete visibility of the transaction path – from the application layer back to the end user
- Unifies all users and traces within a common framework built around "transactions" – tightly coupling data and workflows for seamless collaboration and one-of-a-kind visualizations
- Weaves in the effects of virtualization and shared resource conflicts on web application issues, unifying data and relationships across all transaction dimensions from browser to the database, and from the code down to the hypervisor
- Identifies code-level bottlenecks within Java and .NET application servers with detailed root cause analysis
- Traces the call stack for individual requests, and captures relevant supporting evidence including memory (heap) statistics, method parameters and SQL bind variables
- Leverages this wealth of monitoring data with highly scalable analytics to deliver out-of-the-box analyses and visualizations

Foglight for Web 2.0/AJAX

Foglight captures extensive performance and content data to detect both application performance and functional problems in Web 2.0 applications. This includes:

- Contextual forensics to understand what the user was doing when a problem occurred
  - Capture of keyboard/mouse events for full session context
  - Identification of AJAX events for both errors and performance
  - Identification of functional issues including user input errors
- Accurate page response times and detailed navigation timing
- Consistent round-the-clock testing of availability and performance, even when there is no user activity
- Ability to troubleshoot JavaScript performance and errors
- Ability to isolate problems to specific individual page elements
- Ability to configure the level of monitoring detail within the browser, as required

Conclusion

Web 2.0 technologies, such as JavaScript and AJAX, facilitate faster time-to-market for web applications, while accelerating performance and increasing efficiency. However, they also present several challenges to traditional application performance monitoring.

Foglight from Dell enables DevOps and application support teams to modernize their application performance monitoring strategy and adequately support AJAX processing in the browser, while uniquely combining the best of traditional and modern approaches to deliver a well-rounded APM solution.

Foglight delivers a collaborative customer-centric approach to APM, providing IT and the business alike with answers, not just data.

For More Information

About Dell Software

Dell Software helps customers unlock greater potential through the power of technology—delivering scalable, affordable and simple-to-use solutions that simplify IT and mitigate risk. The Dell Software portfolio addresses five key areas of customer needs: data center and cloud management, information management, mobile workforce management, security and data protection. This software, when combined with Dell hardware and services, drives unmatched efficiency and productivity to accelerate business results. www.dellsoftware.com.

If you have any questions regarding your potential use of this material, contact:

Dell Software
5 Polaris Way
Aliso Viejo, CA 92656
www.dellsoftware.com
Refer to our Web site for regional and international office information.