

Dynamic Clients: The Network Palette Arrives

Net Forecasts – Peter J. Sevcik
BCR Volume 37, Number 7
July 2007

In my May 2001 BCR column (pp. 10-11) I described the emergence of a new application platform that I called the “network palette.” My vision was that content delivery network (CDN) and application delivery network (ADN) services would “evolve from serving as an intermediary that reduces response time across the Net to providing real-time termination of the connection, and become the location where content, processing and support aids reside for each local user.”

I described a new world of “self-propagating applications where applications and content will ‘atomize’ into small, self-contained chunks of useful stuff. So, blow up your website. You’re going to need to get back to ‘home base’ very infrequently. Service will be local.”

But I did caution that in order for this vision to really take hold, “successful platforms also [must] kindle the formation of a community that applies its collective creative talents to bring new capabilities to an open platform. How well a new technology balances proprietary vs. open interests determines whether it evolves from a ‘one-off’ application or product into a true platform.”

Vision Turns Into Reality

Let us now fast forward 6 years. Today, that vision is reality, and it snuck up on us as I predicted--but not as an outgrowth of CDN services. So I was half right.

The poster child for the CDN vision was Akamai. They have grown very successful by serving the needs of their core market. However, they did not open their edge servers to a community that could add functions and features. Instead, they focused their efforts on relentless consolidation of the basic CDN/ADN market by purchasing nearly every competitor in the space. This is a good traditional business strategy of sticking to a single business model, executing well, and wiping out the competition.

In the meantime, the vision of a network palette arrived incrementally on our desktops. It comes in the form of toolbars, extensions, and applets

on top of the browser. Last month, Google announced a quantum leap in this progression when they released Google Gears for Windows.

Google Gears is an open source browser extension that lets developers create Web applications that can run offline. Gears provides three key features:

- Store and serve application resources locally. A local server, to cache and serve application resources (HTML, JavaScript, images, etc.) without needing to contact a central server.
- Store data locally in a fully-searchable relational database. A database, to store and access data from within the browser.
- Run asynchronous JavaScript to improve application responsiveness. A pool of resource threads is established and reserved to quickly execute time-intensive operations.

Google gets it. They are the leading developer and user of what I now call “dynamic clients.” For example, Google Maps, which runs almost completely locally on your desktop, is implemented in Ajax (Asynchronous JavaScript and XML) with incremental feeds of map tiles that only need to arrive in the background. The service adapts to both my office computer and my small Treo.

Dynamic Clients

The Web is undergoing a transition from a collection of autonomous websites to an integrated computing platform serving Web-based applications to end users. This phenomenon, often referred to as “Web 2.0,” is not really a change in any Web standards but rather a shift in how they are applied.

The real change is that the Web application is being deconstructed into separate elements which can reside at the primary website or be distributed among many locations on a network. Each element can then be supplied by a business partner or simply a friend whose website you want to add to your mash-up. Within an enter-

prise this change is driven by Service Oriented Architecture (SOA) and virtualization.

These changes in how applications are being built are supported by many well-known benefits, such as leveraging open source, ability to build applications more quickly, and making the user interface a rich, intimate coupling to the application.

The role of the browser is shifting from a passive client to a dynamic client, as shown in Figure 1. In the passive client mode, all of the presentation logic is loaded to the browser in the base-page portion of every web page. The browser then dutifully gets each page element and paints the pages as instructed.

In the dynamic client mode, the presentation logic has moved to the browser, where software operating above the browser makes choices based on what the user desires.

The dynamic client is a level of software that operates above the browser and within the browser window. Currently common examples of a dynamic client are: Java scripts, Ajax, and Flash. In each case, the user is primarily interacting with the dynamic client that is operating

within a browser tab. In effect, the dynamic client is yet another level up the protocol stack above the browser's Layer 7--which would make it a new Layer 8.

A very important difference in the dynamic client model is that the data required to fill out elements of the Web page no longer come from a single location. Clearly, some of the contents may come from an alternate delivery system like a CDN. But more recently, business logic, customer data, and even the business data that is needed to complete a transaction can come from third party sources.

The browser performs the final assembly of the Web pages based upon a combination of rules defined by the primary Web server, dynamic client, and the user. (The user does not directly set rules, but rather just moves a mouse or makes choices on the screen.) Furthermore, there are many aspects of the application that operate on the desktop that do not interact with any server on the network. The dynamic client is much more autonomous piece of software that is the new true user interface rather than the browser, which is reduced to a supporting role.

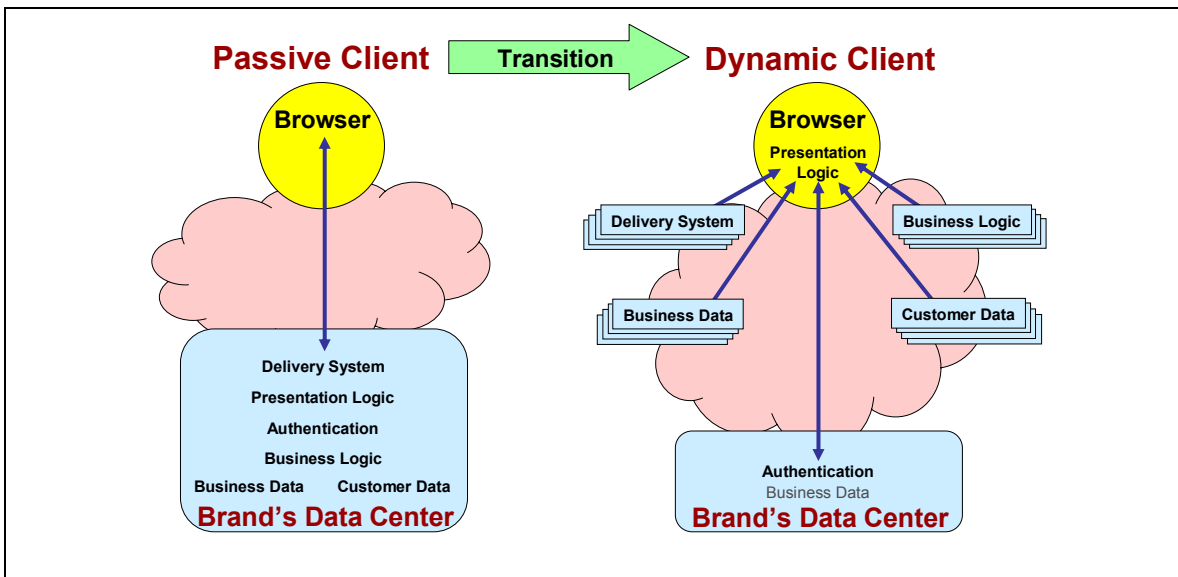


Figure 1 – Shift in How the Web Works

Everyday Examples

Dynamic clients are appearing in many subtle ways. Often, people are operating an application or interacting with a website unaware of the new paradigm. Here are some everyday examples that are occurring today.

Electronics Retailers

The typical on-line store has many more items available than the brick-and-mortar store. This is certainly true of electronics retailers. For example, a major electronics retailer can easily have 1,000 televisions in their catalog. A buyer can search and compare these televisions according to a large number of technical details such as screen size, display type, overall product dimensions, weight, inputs, outputs, and remote features. This is just a sampling of the 50+ technical details available. And they change as each supplier drops and adds models.

The retailer cannot keep current the 50,000 details on all the televisions they sell. So the retailer lets the details about each product come directly from the manufacturer. Critical business information pertinent to a sale appears on the user's browser without the retailer involved.

A major on-line store can easily have 100 such relationships where the content presented to the user comes directly from the manufacturer.

Newspapers

Many newspapers, from local to national brands, offer an on-line "front page" that is customizable by the reader. In fact, the on-line newspaper is an integration of many sources. Again, key elements of the reader's Web experience--like the weather, stock ticker, horoscope, advertisers, social networks and even the newspaper's content search window--come from third parties.

It is often simpler and less error prone to have such content come directly from the source rather than trying to store, update, and coordinate the content from the third parties. A typical newspaper can have such direct data feeds originate from dozens of outside sources.

Banks

Most banks provide an on-line account management capability. The information regarding your bank balance, checks cashed and on-line payments comes from the bank's computers. But many banks provide many additional services as they try to be the single place to go for all your financial needs.

Some banks provide an interesting bill presentation service. For example, you can link your electric bill directly to your bank's on-line interface. This is not the ability to have the electric company automatically withdraw a payment from you checking account each month. Instead, this is the ability to have you review your electric bill from within the confines of your banking session.

When you are switched to the electric company, that portion of the bank's Web page is handed over to the electric company. The session, along with the bank's passing of your authentication credentials, is what gets handed over to the electric company. Without you needing to do any other steps, you are now in a direct dialog with the electric company systems. You can review your electric bill and choose to pay some, all, or none. The amount you decide to pay is passed back to the bank and you continue with the bill payment process.

The critical point is that for a period of time, your secure SSL session was passed to a third party. The bank had no view of what transpired between you and the electric company. Similar secure hand-offs occur for purchasing insurance or switching to the bank's brokerage partner.

Your Brand Is On Top

In the above examples, the primary brand was the retailer, newspaper, and bank. When some portion of the Web page was being supplied by one of the partners, the brand on the top of the Web page was that of the retailer, newspaper or bank.

This means that if something goes wrong--if content is not delivered, or performance slows to crawl inside the partner's window--the user still thinks he is talking to the primary brand. The real danger is that your brand's reputation is at a heightened risk, yet due to the nature of the dy-

dynamic client your control has diminished. Higher risk with less control is a recipe for disaster.

Summary

The revolution of dynamic clients is here. What Google did with Google Gears is one more step to make it simple for any Web developer to use. The number of applications that operate as dynamic clients will explode.

Google is clearly bringing us the vision I described six years ago. It is interesting that when I first described the vision, Akamai and Google were very equivalent companies. Both were exactly 3 years old and both had the same annual revenue. Today Google's revenue is 25 times that of Akamai. Executing on the vision to really move applications to the user paid off.

Of course, this brave new world will bring with it new challenges to manage all the locally running applications, coordinate with parent websites, ensure end-user experience performance, and avoid client conflicts. But that is the price of success. Future columns will address new tools and services that are emerging to address these opportunities.

Companies Mentioned

Akamai (www.akamai.com)

Google (www.google.com)

Peter Sevcik is president of NetForecast and is a leading authority on Internet traffic, performance and technology. Peter has contributed to the design of more than 100 networks, including the Internet, and holds the patent on application response-time prediction. He can be reached at peter@netforecast.com.

NetForecast is an internationally recognized engineering consulting company that analyzes and improves data, voice, and video application performance. We help enterprises align application performance with business needs using a process based on the Apdex standard. NetForecast also advises technology vendors about customer requirements, technology issues, and the business value of application delivery products and services. Visit our library of educational reports and articles about performance engineering.

www.netforecast.com

