Web 2.0 and Mashups:
Where Do We Go from Here?

Unconventional Wisdom
Tom Nolle, Founder and President, CIMI Corporation

Introduction

The notion of the next generation of anything is appealing, and when the "something" is the Internet or the web it's doubly so. Thus, the notion of "Web 2.0" is something that's been promoted to the point of becoming a cliché. We've had just about everything-two-point-oh at this point, in fact.

To add to the confusion, the whole Web 2.0 thing is far from nailed down in terms of definition. There are hosts of people who say they have it, or use it, or provide it, but somewhat of a shortage of definitions. Even where you see definitions these days, you often find them rather…shall we say…philosophical.

"Web 2.0" has come to be used (perhaps from the very first) as more hype or as a means of goal-expression than reality. The idea arose at a conference in the early part of this decade, and one could fairly say that the notion was that Web 2.0 was whatever you thought would make Web 1.0 more palatable, more exciting, more hype-worthy. If you are going to talk about the impact of Web 2.0 on networking, media, advertising, or anything else, you have to somehow shed the fluff and move to something objective. We propose to do that, and of course will likely anger many of the proponents of the concept. Hey, differences of opinion are what make horse races, right?

Despite our diving downward past the glitter, there are a lot of technical details below the level we're going to cover here, of course, and that's the sort of thing that will lead many to dismiss the effort we're making as "incomplete". OK, but unfortunately there are books about each individual aspect of Web 2.0, which should make it clear that covering it in five to seven pages will require a bit of creative prioritization and condensation.

Starting with One (of *N?*)

The web as it was first developed was a combination of a formatting language (HTML or hypertext markup language) and a protocol (HTTP or hypertext transfer protocol). The essential idea of the web was to transmit an HTML document (a web page) to a client (the browser) where it would be rendered as a pretty visual display that could include images and other good stuff. The document would usually include hyperlinks to other documents or "pages" and clicking on one activated the HTTP process to obtain another document, and the process repeated itself. You click a URL to load a page, and the page presents other URLs, which if clicked load other pages.

This notion, which we'll call "Web 1.0" in large part because you have to define the first phase of something to even approach the second, had strengths and limitations. The whole concept of a hyperlinked set of documents that could be browsed by clicking creates intuitive navigation and (for the time, the early '90s) a rich experience for the user. But the HTML/HTTP thing was limited in that it

created a very rigid model of experience. A document in HTTP format, once loaded into the browser, created the experience the user obtained. If you wanted to see something different, you loaded another document into the browser. Load page. Click link, and load another. It was all or nothing. Imagine the process of multimedia in such a situation; you'd almost have to load successive images the way that early "movies" were created, by flipping cards quickly. That worked in the days where visually "moving" images were rare, but it got old fast.

So it is with Web 1.0. Obviously it took a bit of time for the insatiable appetite of users for pretty stuff to exhaust the capabilities of Web 1.0, but it was pretty clear early on that this would happen. A number of enhancements were introduced to try to make things better, and these are still used by many (maybe most) sites today.

One simple enhancement is the notion of scripting, and the use of browser "applets" that can do things on their own. This can be added to HTML to create what you could call an "active page", meaning that the page had logic in it that did stuff and could create what you might call "sub-experiences" without demanding that everything be replaced from the server. Client-side scripting using JavaScript, ActiveX, etc. has added a lot to web pages' appeal, and has also provided a way to create what is essentially a small distributed application if the client logic can interworks with the server logic.

Another enhancement is the notion of an "add-in" or "plug-in" that is associated with a particular type of file. Browsers have built-in capability to view things like GIF or JPG files, but with additional "viewers" they can open other files as well. Many people have seen Adobe PDFs in a browser window—that's a viewer plug-in. Almost everyone has seen Flash or QuickTime videos, also supported by their own plug-ins.

All of this is good stuff, but all of it also operates in a restricted world where the HTTP/HTML relationship sets the overall context of everything. You can really only send one thing via HTTP, and that's a request for a page to be loaded. If what you want to do can't be managed by that simple client/server interaction, you have a problem. The newer interactions, ones that permit events **within** a page, without reloading, create the new and critical foundations that we'll call "Web 2.0" because they are the framework that makes any of the higher, fluffier, fuzzier, goals possible.

Imagine a browser as a gateway into experiences, whether work or entertainment. The old page-click-page concept meant that the gateway could show you only one place at a time. If a page were created by multiple contexts, multiple experiences, then the owner of the page had to do the coordination and compositing because any click replaced **everything** on the page. The challenge was to create something that let a page be "zoned" and controlled independently, something that made pages truly interactive.

Work on something to allow for more complex page interaction began before what we'd call the "Web 2.0" period. Microsoft and Netscape both had "remote scripting" languages that created intra-page behavior control. The evolution of this client-script approach eventually resulted in what is broadly known as Dynamic HTML or DHTML. Microsoft evolved its own solutions using .NET, and Macromedia (now Adobe) had the same with its expanding Flash player capability set. All of this stuff worked, but we would assert that the critical step came with the concept called "Ajax".

Ajax (sometimes also spelled "AJAX") is a rather cute idea. It's a combination of a browser client-side component and a server-side component (Java is presumed but other stuff also works) that communicate via XML messages. The client-side element controls the local interaction to the user and the server side

fields messages that ask for content, processing, etc. If you've surfed the web at all these days, and in particular if you've visited retail sites or social sites, you've almost certainly encountered Ajax.

The big advantage of the Ajax approach is that you create what is in effect an overlay or layer on top of a standard web page, and this can be manipulated via Ajax. The overlay can deal with forms and content underneath, so you still have page-related activity, but the Ajax process is asynchronous to the normal web refreshing. This creates one of the disadvantages; Ajax pages don't support the "back" button well because hitting it doesn't refresh the Ajax dialog, it refreshes the page.

The client/server model used by Ajax also represents pretty accurately the "viewer" or "player" model of interaction we've described, and perhaps most importantly it's also how Microsoft's Silverlight works. Silverlight, and Adobe's Flex architecture that will compete with it, likely set the stage on how "Web 2.0" will evolve at the technical level. Adobe has recently released AIR, which is a kind of Flex virtual machine that allows applications to be deployed purely local, purely over-the-network, or a hybrid of the two. This matches what Silverlight/Vista will offer and beats it in the near term, and so blurs the distinction between Web applications and local applications. Since AIR isn't a part of an operating system but an independent local system runtime, it could truly change the way that applications are written in the future, making on-web and on-desk almost irrelevant. That's one of the early "hype" characteristics of Web 2.0, and so we think the technical trends we're citing here are validated as the fundamental ones involved in the Web 2.0 transition.

AIR brings local applications to mashups, creating a powerful harmony of web and desktop applications that could really improve productivity, and along the way enable a multiplicity of new service/application models. When AIR was first announced we said then that it would bring a fundamental change in how applications work. Now, eBay has announced AIR-based desktop applications, and AIR isn't even yet in a production release. In our view, Adobe's AIR is the technology to watch in this space, combining flexibility, mobility between online and offline, and portability to many different platforms to create a truly awesome package. It's not unreasonable to say that AIR or something AIR-like might, in the future, end up more important than an operating system.

The Silverlight and AIR approaches essentially thicken clients to the point where they begin to look like independent application components. This combines with the shift toward asynchronous, interactive, client/server technology for web interaction to generate much less "web-like" behavior. One wag suggested that the most defining truth about Web 2.0 was that it wasn't really the web at all.

Where Does Web 2.0 Lead Us?

Which begs the question of what it **is** like. There are many who decry the current trends since they create pages that aren't really searchable in the classical sense, and may in fact create a drive toward sites that are not really even universally accessible. Will everyone load Flex or Silverlight? To many, the debate smacks of the broader debate on the mission of the thing we call "The Internet".

The problem is that the Internet hasn't been profitable, and those seeking profit have to seek something exciting and differentiable that you can do **on the Internet**. That leads us to the Web 2.0 technology, like it or not. Increased bandwidth created by the consumer broadband movement creates an opportunity to deliver a richer and more interactive experience, and that sells those above-the-net applications that are

keeping the lights on in Silicon Valley and elsewhere. Web 2.0, meaning more visually interesting and interactive/engaging sites, is an inevitable development.

Perhaps the single key shift that Web 2.0 can/will bring about is the shift from a homogeneous page to a mashup. The concept of mashups is directly linked to Web 2.0 because without it the creation of a single "viewport" from a diverse set of inputs would have to be done by having someone organize the views and present the result, a concept sometimes called "screen scraping" because it removes pieces of content from a variety of screens and gloms them into a single view. With Web 2.0, pieces of a page can be laid on a single foundation providing that the contributor of the pieces uses compatible mashup formats.

Mashups are the key to personalization, which of course is at least a big step toward engagement, and they're important both for the consumer market and for the enterprise. Mashups are the number one path toward productivity enhancement according to CIO types. For the consumer, they can be used to create personalized portals that draw from interest areas spread all over the place.

It is very likely that mashups will create a separate layer in information presentation and site design, with the "main layer" remaining the normal website, enhanced in interactivity via Web 2.0 but still representing a single vision of information. The new layer will be "mash-able" meaning that it will be something like a widget/gadget that presents a specialized set of information into a flexible context. Some sites may compose even their main layer via mashing, but we think that for the most part and particularly in the near term there will be a clear distinction between a normal web page and a mashup component.

The introduction of mashups into the process of web-building complicates the whole picture of IP services. First, it is clear that data on a "page" will arrive from multiple destinations, and that the unity of experience will require that each destination fill its portion of the viewport promptly. Second, it is clear that there will be interreactivity possible between mashup components, arbitrated perhaps by the underlying page. This means that there will be an opportunity to "own" the viewport and arrange the mashups, creating a new possible mission for portal vendors or even a new class of players.

For many applications in both the consumer and the enterprise space, we expect that mash-able components will be the only interfaces, and we also expect there to be increased interest in creating harmoniously interworking components to mashups. This process can be done easily (relatively) from the server side of the process, and so it may create a new class of tool and even a new kind of web page, a "mashup host" page. Our research shows that this kind of process could be viewed as a SaaS application, meaning that enterprises might host the composition of mashups. However, it appears more likely that a component of mashups, such as a collaborative tool, might provide both the mashup host page and a key mashup component, and thus become the focus point for the mashup. Google and Yahoo clearly have this sort of thing in mind; we think Yahoo Pipes is perhaps the clearest example of how this might develop that's currently popular.

How mashups would impact the concept of portals or "eyeball ownership" is hard to say. As we've indicated, the concept of a host page might introduce a new way to claim eyeball share or a new set of competitors—likely both. The atomization of information via mashups creates a kind of best-of-breed mindset that could make it hard for someone like Google to grab all of the search market, for example, or to use searching to extend its influence into related areas. It might well encourage the development of richer information tools and richer pages with a few "mashup slots" to encourage users to rely mostly on the portal players for information and to supplement only to a limited degree. There is already work on finding out how many mashups a given user might want at a given time.

There are real implications to this model of information presentation because with mashups it is not at all easy to see how searches could be conducted. If there is no such thing as a "page" then what is indexed? If you find a mashup in a search, how do you invoke it or compose it into the view of the finder? Mashups could create a multi-leveled process of indexing where underlying content is indexed but not presentation layer information, but if that happens how does the information "found" get presented, since it's not on a classical, viewable, page?

It's the glimmers of problems like these that have the Internet purists worried, and it does appear that there is a major shift about to happen in the way that users relate to the web, and that information is then conveyed to the user. The impact of this shift could be profound, significant enough to unseat some players in the industry and to elevate others. It also creates an enormous opportunity in the gray area of information management between the viewer and the storage process.

We wonder who will see it, and strike.

About the Author

Tom Nolle is the founder and president of CIMI Corporation, a telecommunications, media, and technology consulting and research firm since 1982. Tom is a software architect and developer by background, but currently focuses on strategic market, product, financial, and regulatory issues. He's also the Chief Strategist for ExperiaSphere, an open source initiative that focuses on development of Java-based service logic and service management frameworks for next-generation networking. Tom is a contributor to Network World, the No Jitter CMP publication, Telecommunications Magazine, Tech Target Search Telecom, and Internet Evolution. He's also the author and publisher of Netwatcher, the oldest continuously published networking publication in the industry. He's a 25-year member of the IEEE Communications Society, a member of the Telemanagement Forum, and a speaker at many industry events.

About the Webtorials® Editorial/Analyst Division

The Webtorials® Editorial/Analyst Division, a joint venture of industry veterans Steven Taylor and Jim Metzler, is devoted to performing in-depth analysis and research in focused areas such as Metro Ethernet and MPLS, as well as in areas that cross the traditional functional boundaries of IT, such as Unified Communications and Application Delivery. The Editorial/Analyst Division's focus is on providing actionable insight through custom research with a forward looking viewpoint. Through reports that examine industry dynamics from both a demand and a supply perspective, the firm educates the marketplace both on emerging trends and the role that IT products, services and processes play in responding to those trends.

For more information and for additional Webtorials® Editorial/Analyst Division products, please contact Jim Metzler at jim@webtorials.com or Steven Taylor at taylor@webtorials.com.

Webtorials Briefs
Vol 2, Number 11

Published by Webtorials Editorial/Analyst Division
www.Webtorials.com

Division Cofounders:
Jim Metzler
jim@webtorials.com
Steven Taylor
taylor@webtorials.com

Professional Opinions Disclaimer
All information presented and opinions expressed in this publication represent the current opinions of the author(s) based on professional judgment and best available information at the time of the presentation. Consequently, the information is subject to change, and no liability for advice presented is assumed. Ultimate responsibility for choice of appropriate solutions remains with the reader.

For editorial and sponsorship information, contact Jim Metzler or Steven Taylor. The Webtorials Editorial/Analyst Division is an analyst and consulting joint venture of Steven Taylor and Jim Metzler.