AVAILABLE MARCH 2007

WEB 2.0

IN SIMPLE STEPS

AVAILABLE MARCH 2007

HAKEEM SHITTU

genix

CHAPTER 1

A BRIEF HISTORY OF THE WEB

The web is changing. If you have been paying attention these last several months you would have begun to notice a definitive change in the appearance and usage of the web. Perhaps you became aware of this change with sites such as Wikipedia (wikipedia.org) where you could collaborate with other individuals to provide content useful to other users or through one of the numerous sites that remixed content from Google Maps (maps.google.com). Several sites with similar ease of use and high level of user interaction continue to appear everyday on the web. In fact, in some cases, the boundary between a desktop application and a web application has virtually been erased. You don't think so? Well take a look at Writely (writely.com) or Zimbra (zimbra.com) as an example of sites that have blurred the line between the web and the desktop. Without the browser bar on your screen, you might forget that you were using a web based application, and assume that the software in use was installed on your computer system. These sites, and other similar ones that you have been seeing, are signs of the coming of Web 2.0, a movement that seeks to evolve the web from being simply a document delivery mechanism into a platform for collaborative activity and expanded application functionality.

The term Web 2.0 is credited to Tim O'Reilly, the publisher of O'Reilly books, who coined the moniker in a bid to describe the new generation of services that were appearing on

the net which he thought had the potential to revolutionize how people worked, viewed data and shared information over the internet. To reduce this definition to the simplest terms, we can regard Web 2.0 as the next generation of the web. However, unlike software applications that have a specific release date, there is no date for the release of Web 2.0. It is expected to gradually work its way into our existence as users flock to services that adopt this platform and businesses realize the profit potential of these sites.

To understand what has been added to the web to create Web 2.0, we must take a little trip down the evolution of web technologies to see what type of architecture existed in the earlier days of the web and how the supporting architecture has changed in recent years. We will also look at the evolution of presentation technologies that were added on to make HTML and our web browsers more interactive and feature rich.

EVOLUTION OF WEB ARCHITECTURE

The first architecture created to support the retrieval and transmission of documents on the world wide web was a client-server model. In this model, a client would connect to a server and retrieve documents available on that server. The client would then display these documents on its screen. Besides having a web browser, there was no additional software that the client needed to install. This proved to be very useful and it gave users the ability to browse through content that was available on the web. The functional uses of an information dispersal system of this nature include such actions as the delivery of news, the dissemination of information about a company (for example through a corporate web site) and product advertising.

The web proved to be very successful early on, and the ease of navigation as well as the graphical layout of the content won over many user. Two of the technical components of the web that are often credited for the early success of web usage include:

HTML - Documents sent by a web server are based on an open, standardized document markup language known as Hypertext Markup Language. This language allows users to create a document marked with special tags that determine the formatting of the document. When the documents are retrieved and rendered inside any web browser, they look exactly like what the page creator designed, making this a desirable vehicle for disseminating documents. Some of the features of HTML that contributed to the languages' early success include:

- Hypertexting – The ability to jump to a different resource by selecting a 'hotlink' that exists on one resource. Hypertexting truly provided users the ability to 'browse the web' as users were able to move from one related document to the other without having to know the names of these resources.
- Standards based - Allowing developers to create browsers for different operating systems and devices that would render a HTML page in the same format as the page creator designed the page.
- Platform independence - HTML allowed users of different operating systems to view the same information as long as they had a web browser installed. This way, web page designers did not have the burden of trying to support several different operating systems.
- Human readable - Using a human readable text format allowed people to borrow interesting design concepts from several sites and use it in their own.

NOTE: This remixability is an early form of what is later known as a mashup

URL - Uniform Resource Locators are a simple way to uniquely locate a document across the vastness of the Internet. The URL contains specific parts which, when combined, would uniquely identify a resource. The main parts of a URL are as follows:

protocol://host:port/file

protocol
: The communication protocol that will be used to access the server. This is typically http or https.

host
: The DNS name of the host to connect to or the IP address of the host.

port
: The port address on which the web server is listening. When the port is not included, the browser sends the request to port 80.

file
: The fully qualified path name to the resource being accessed on the server.

NOTE: A virtual filesystem is a directory on the web server that is designated as the root directory for the web server.

An example of this URL would be http://www.genixpress.com/index.html where the HTTP protocol would be used to access the web server listening at genixpress.com on port number 80. The request is for the file named *index.html* existing at the root directory of the web server's virtual filesystem.

Two-tiers

The initial client/server architecture of the web served web users well. Figure 1.1 shows the elements involved in this client/server architecture. This design is sometimes referred to as a 2-tier architecture with the presentation tier being available on the client while the data tier existed on the server.

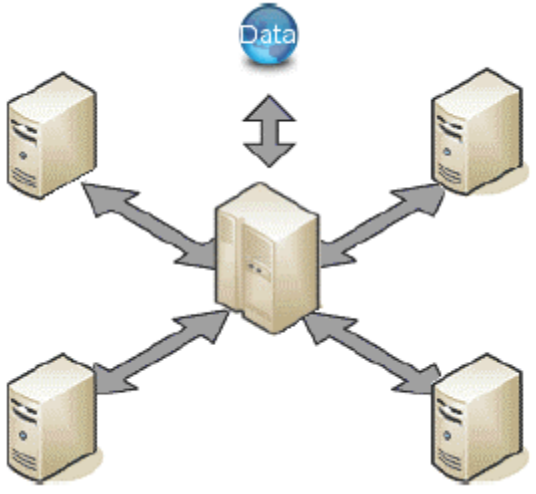

Figure 1.1: Client-server architecture model

As shown in Figure 1.1, the web browser (also the client) is installed on the user's system. When the user types in a URL into their browser window, or clicks on a hyperlink, the browser initializes a request to the web server for the document. The web server retrieves the requested document from its filesystem and then responds to the client by sending the requested document. On receipt of the document, the client would immediately display a visual representation of the document within the browser. Figure 1.2 shows this interaction between the client and browser.

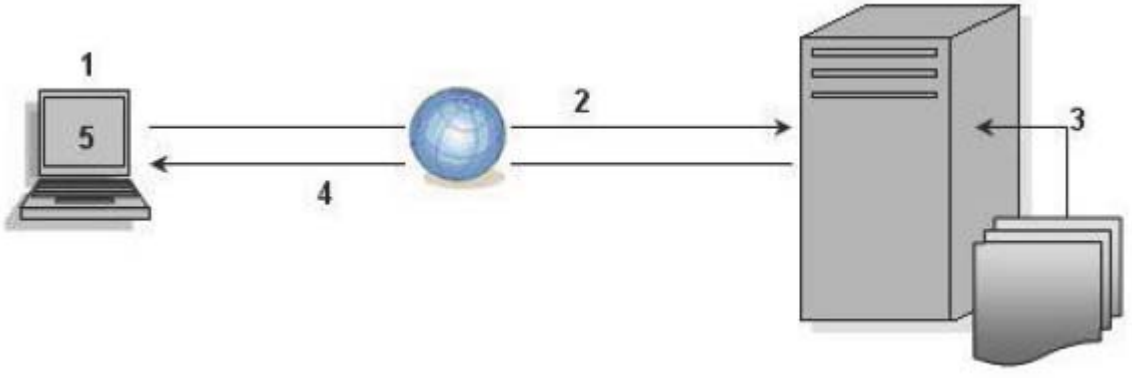

Figure 1.2: Client-server interaction diagram

I have included numbers below to describe the sequence of events that happen from the initiation of the request to retrieve a file from the Genixpress website until the fulfillment of that request.

Activity: A person intends to view the books available on the Genixpress web site.

1. Client initiates a request by typing the URL http://www.genixcorp.com/index.html into their browser address window.
2. The request is routed to the genixcorp.com domain and is received by the web server listening on port 80. [Note: Port 80 is the default port used by the http protocol unless an alternate port is specified.]
3. The web server receives the request, and retrieves the index.html file that is identified in the URL from the filesystem.
4. The web server responds to the client by sending the index.html file using the same

http protocol specified in the request URL.

5. The client browser receives the page and renders it within the browser to display the Genixpress site.

Result: The user is able to view the contents of the site.

The limitations of a 2-tier system became more evident as users wanted to do more with the web than just browse pages. There was such a wealth of information available that the need to search through data was soon necessary. Users wanted interactivity and a need for the web to provide additional useful features. An architecture upgrade became necessary.

Three-tiers

With the need to have web pages support programmable logic, the two-tier architecture needed to be extended. This was expanded to include an application logic tier on the server side. The application logic tier contains a software application that would receive incoming input, decipher it and perform the appropriate action(s) requested by the user.

Search functionality, for instance, is supported by such an architecture as this. User input to the server is read by a web application which then processes the search term, accesses the data layer (which now exists separately) and returns the resulting information to the user. Figure 1.3 shows this interaction.

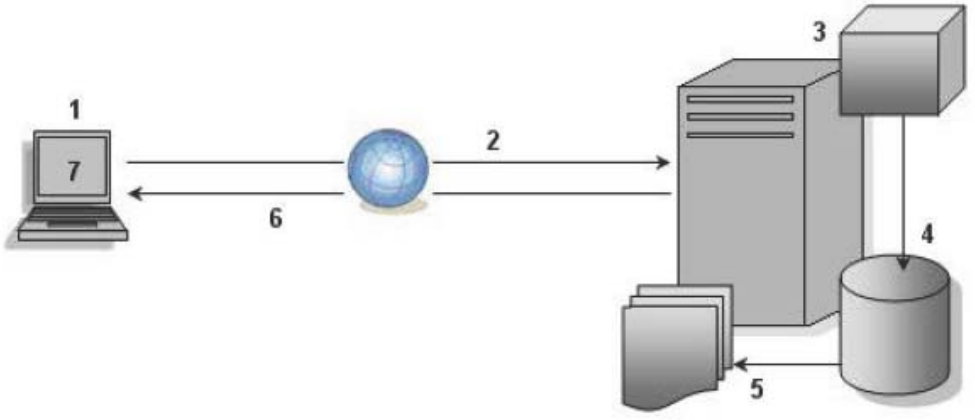

Figure 1.3: Interaction diagram of a multi-tier based system.

The earliest, simplest technology that was used to make this happen was Common Gateway Interface (CGI), a standard protocol that allowed developers to create software in any language and then connect these programs to their web servers using CGI.

Activity: A person intends to search for the current list price of items on the auction site eBay.

1. Client initiates a request on ebay.com by typing an item search term of "PS3" into the available search field.
2. The request is routed to the ebay.com domain and is received by the web server listening on port 80.
3. The web server receives the request, and invokes the CGI application specified in the URL. The additional details needed by the CGI application are encoded in the URL

stream that was sent to the server by the client.

4. The CGI application connects to its datasource and searches within it for the search term "PS3".
5. The CGI application collates these results and formats it into an appropriate HTML page.
6. The web server responds to the client by sending the generated HTML page back to the client using the same http protocol specified in the request URL.
7. The client browser receives the page and renders it within the browser to display the list of items matching the search term.

Result: A page is displayed with a list of items matching the search term.

Because CGI exists as a glue between application software and web server, many people felt that CGI was limited in the functionality that it could support. This led way to the expansion of existing languages to provide native support for web programming. Allaire introduced ColdFusion, Sun unveiled the Enterprise Edition version of its Java programming language, Microsoft unveiled ASP which would later evolve into ASP.NET etc. This list is by no means meant to be exhaustive but showcases a shift among programming languages to provide built in support for web application development. For web applications created using these languages, you will more commonly hear the web server referred to as an application server. That is because servers supporting the web applications do a lot more than just process web requests, they are hosting complete applications and have to execute the applications, manage session state, run threads etc.

The three tier architecture was sufficient for many applications, and still remains adequate for some, however, a proliferation of services now exist online such that certain businesses need to interface with many additional systems in order to meet their user's needs.

Using a search site such as Google as an example, users might need to search through information written in several different languages (English, Spanish, German etc), created in several different file formats (PDFs, PowerPoint slides, HTML pages etc), and have this information available to them on a host of device types (PDAs, computers, cell phones etc). The services requested from many web applications have grown sufficiently large to necessitate an upgrade of the architecture.

_n_-tiers

The n-tier architecture was created to cater to the need for some web applications to interface with an unbounded number of systems to either perform:

Presentation

Visually generate a page based on the device type and/or browser type that initiated the request.

Application processing

Connection to one or more systems to perform intermediate processing in order to satisfy a request. Shown in Figure 1.4 is an n-tier web application for a search engine.

Data access

Needed data could exist in a variety of data sources (databases, files, computer systems) spread across geographically distant locations.

Figure 1.4: The architecture of an n-tier web application for a search engine.

As shown in Figure 1.4, the term n-tier is the only descriptive term that can be used to describe such a multi-connected system as is implemented in many web applications.

It should be noted that Web 2.0 does not require any further change to the architecture of web applications. The *n* tiers of the n-tier architecture already accounts for collaboration with an indeterminately large number of systems. Most applications existing on a 3-tier system would probably have to move to an *n*-tier architecture, but this is dependent on the complexity of the individual application.

Now that we are familiar with the overall architecture of web applications, let's examine the evolution of client side technologies used in the web.

EVOLUTION OF CLIENT TECHNOLOGIES

As I have said earlier, one of the features that were found most useful to the web was HTML, a simple and standards based format for creating web pages. A limitation of HTML is that it is minimally interactive and so early on, technologies were sought to add logic and interactivity to HTML without breaking the platform independent nature of the language.

Some of these add-ons to HTML are included below in the order of their release.

JavaScript

This creation of Netscape was designed to provide interactivity to web applications. JavaScript code is downloaded by the web browser and executed within the web

browser to perform such support functions such as validation and image manipulation. The language became widely adopted because it provided rich programmatic functionality to the web page with posing a security threat to the user.

VBScript

VBScript is Microsoft's implementation of a browser-based scripting language. Similar in concept to JavaScript, VBScript can execute within a web browser and provide functionality to the browser. However, only Microsoft's Internet Explorer provided VBScript support which severely limited the number of sites created with VBScript as the scripting language.

Java Applets

Java Applets execute within a web browser on the client machine providing complete application functionality to the client without the need to install additional software. Applets provided a novel way of delivering applications that could run within a browser without interacting with the local filesystem.

ActiveX Controls

ActiveX controls provide fully functional applications hosted within a web browser, providing the user with the richness of a web application within a browser interface. However, ActiveX controls are only usable within Microsoft operating system environments and even then they are not supported by all browsers. This limits the number of potential users of the application.

Flash

Flash technology is another format for providing complete applications to a user through a browser interface. Because of its widespread support within web browsers on many operating systems and devices, Flash is heavily used by many web sites.

The flexibility of this technology allows Flash application to be used for such items as intelligent web ads and not only complete applications.

Cascading Style Sheet

CSS provided developers with a richer language for describing the presentation of documents. With CSS, the visual presentation of applications could be easily targeted to the type of device that was displaying the application giving developers and designers the ability to utilize the optimal presentation capability of each device.

CATALYSTS FOR CHANGE

Every morning as I prepare for work, listening to podcasts of my favorite shows while checking the local traffic for my morning commute on my Tivo, I can't help but imagine that the web has evolved beyond what anyone could have imagined in its formative stages. As such, there is a need for the web to expand to seamlessly support the new functionality required by today's tech savvy public. On close inspection of the driving forces behind this need for change, I have found that they can be broken down into 3 areas.

Technology

Technological innovations are one of the major driving forces of Web 2.0, technical barriers such as speed and accessibility that had previously limited what could be done have been sufficiently overcome. Some of these technological considerations driving Web 2.0 include:

Wireless

A trip through any area with a concentration of young people will clearly confirm a certain trend - people want to be connected, at all times, without having being

tethered to a desktop system. The use of laptops has sky-rocked with **Wi-Fi** quickly gaining its (well deserved) place in popular language. Connected devices based on this and other wireless communication technologies such as Bluetooth are commonplace and are widespread enough that affordability is less of an issue.

Browser Technologies

The web browser has quietly matured in recent years from a document delivery mechanism to a software platform capable of hosting a variety of technologies including script-languages, Java applets, Active-X (COM) objects, Flash applications etc. These technologies help provide the rich-user experience that is sought in Web 2.0 applications, allowing users not to worry about the difference between desktop and web based applications.

Flexible Data Formats

The role of XML in advancing web technologies cannot be overstated. The advent of the XML data format and the corresponding technologies such as DTD, XSD, XSLT, XPath etc allow companies to view data as a distinct product from presentation. Companies that specialize in content can then continue to create a sell content while those that specialize in presentation can use purchase/leased content to create useful applications.

Social Trends

The community of web users and developers are seeking to change the web to suit how they live, work and play. These individuals are creating applications and contents to support such activities as blogging, social networking etc. Some of the main social catalysts for changing the standard web include:

Broadband

We are a long way past the initial start of the web when the raucous connection sound of a dialup modem filled many homes. Today, a majority of household with internet access make use of broadband connection. This allows more information to flow quickly to and from a server, giving developers the opportunities to use appealing images and other such content on their site.

Generation Web

An upcoming generation of users that have always 'lived' on the web are getting to the age where they are challenging existing boundaries. These users see the web as outlets for messaging, networking, interaction, learning, employment etc and need to have applications that support these uses. Where none exist, they have begun to design and create such applications that gel with their vision of the web.

Mixability

Remixing is not for music alone. Users want to use existing data and software in creating application and content that is relevant to them. The myriad of Google Map hacks that sprung up to utilize the Google Map API as well as the large number of users of the GreaseMonkey (greasemonkey.mozdev.org) extension for Firefox browser show that users want to 'customize' just about everything they can.

Business Needs

Businesses have long needed a way to tap into the large number of people in their audience to advertise, suggest enhancements for and distribute information about their products. Some of the main needs for companies that are bring about Web 2.0 include:

The Long Tail

Chris Anderson, in a Wired Magazine article described the model of doing business online successfully as "selling less of more". Statistically represented in a pareto chart

as an elongated tail (and shown in Figure 1.5), the Long Tail, can be simplified as a business makes more money by selling several low volume items than a few blockbuster items. A book store for instance, makes more money selling a few copies of a lot of different books every than from occasional blockbuster books. This model is displayed by such successful firms as Amazon, eBay and Netflix.

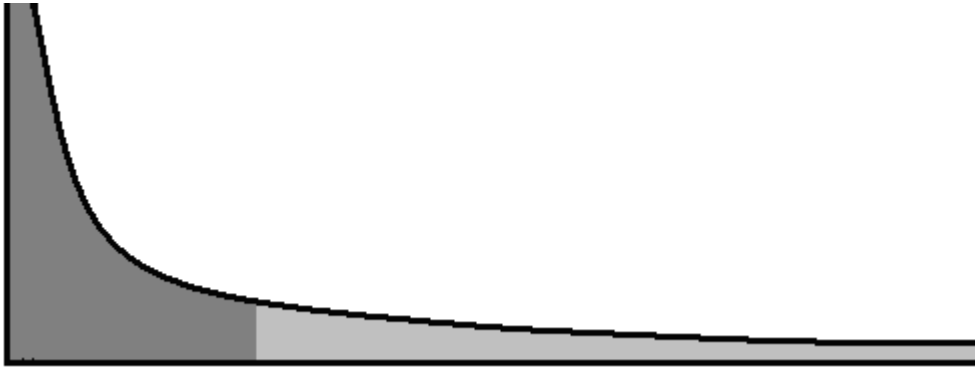

Figure 1.5: The Long Tail

Viral marketing

Like tupperware parties of yesteryears showed, the most effective way for companies to succeed with a product is to get a group of like minded individuals together. Concepts such as social networking is making it possible to spread word about a product through a network of trusted individuals.

Collective Intelligence

It turns out customers are not just content to consume products, they also wish to

generate information about these products such as reviews, alternate uses and suggestions for enhancements. Amazon (amazon.com) found out early that customers preferred reviews written by individuals that have purchased and read a book than by critics. Businesses are looking into ways to tap into this army of freelancers and these have led to the creation of useful Web 2.0 concepts such as wikis and folksonomy.

Web Services

The ability to sell data as a standalone product without the need of packaging it into a presentation interface has considerably transformed the business world. Through the use of web services, Company A is able to make money by providing Company B with access to it's (Company A's) data. Company B can then build a suitable application to make use of the data and sell this to customers. This way, Company A and Company B are able to concentrate on their individual core competencies.

WHO IS IMPACTED

It should not be seen as being too self indulgent to say that Web 2.0 will have a significant impact on everyone. Look through your daily operations and imagine the operations that you rely on a computer to deliver to you and which only years ago were delivered through a different medium. Driving directions is one such item. When was the last time you paid for a bill using a check or took a trip to your bank to verify the balance in your bank account? A few years ago it was inconceivable to imagine that the way we performed these standard activities would change, yet, we are on the verge of another movement that could alter these information even more.

The power of Web 2.0 is that it will revolutionalize how we consume information. Some projects in current usage are beginning to show us how this will impact our lives. Projects

such as Wikipedia (wikipedia.org) are changing how we learn about new material by using a collaborative model to edit massive entries of data into the online encyclopedia. Sites like Digg (digg.com) are changing how we read news by creating a medium where users aggregate and then vote on specific news items. Though many of these sites provide the rich user experience that is synonymous with Web 2.0, you may notice that this is not a requirement. The commonality of these projects is that they are altering how we do things.

SUMMARY

This chapter provides an introduction to Web 2.0. The chapter begins with a discussion of the gradual move to Web 2.0 being seen among some heavily trafficked sites on the internet. I described the evolution of the web architecture upon which web applications are hosted and also discussed the growth of client side technologies that we use to access these applications. This should help you to understand the current capabilities of the web.

Next, I stated the underlying factors that are a catalyst for Web 2.0 movement identifying technological, social and business factors that are aiding in this process. After digesting this introductory material, we are now ready to move on to the specific contents available from Web 2.0. That will be discussed in the next chapter.

KEY TERMS

ASCII American Standard Code for Information Interchange, is a character encoding based on the English alphabet used for representing text in computer systems.

CGI Common Gateway Interface, a programming standard that allows web servers to interact with programs.

DNS name A name leased for use from an internet name registry that allows users to connect to a host using this alias instead of remembering the machine's IP address.

DNS name	A name leased for use from an internet name registry that allows users to connect to a host using this alias instead of remembering the machine's IP address.
HTML	HyperText Markup Language, is a software language that is used to format document so that they can easily be transported over a communication medium and displayed as intended by the creator. This is the language used for creating web pages.
IP address	An address consisting of the following numbering scheme: 0-255. 0-255. 0-255. 0-255 assigned to a networked computer that allows other computers to communicate with the machine.
URL	Uniform Resource Locator, an address that uniquely identifies a file on the Internet.
WI-FI	A communications technology allowing devices to wirelessly connect to each other or to a local area network (LAN).
XML	Extensible Markup Language. A file format for transporting data in text format while maintaining the integrity of the data.