# Web Development in the Graduate IS Curriculum: Mashups – Web Services and Hybrid Applications

Narayan Murthy
nmurthy@pace.edu

Daniel Farkas
dfarkas@pace.edu

Seidenberg School of Computer Science and Information Systems
Pace University
861 Bedford Road
Pleasantville, NY 10570

**ABSTRACT**

As more Information Technology systems exploit the Internet with existing web services, it is important for IS graduate students to be able to understand the development of web-based applications. The latest IS model curriculum for graduate information systems emphasizes the utilization of XML and web services as part of the IS Technology core. This paper describes a graduate course which teaches students how to create "mashups". Mashups are web applications developed by using two or more existing applications or services. Well known companies such as Google, Yahoo, Amazon.com, EBay and others provide services to developers which are used with application specific Application Programming Interfaces (API). This paper gives a brief overview of mashups and an example of how one is created.

**Keywords:** Web Development, Web Services, Graduate Curriculum, XML, Mashups

## 1. INTRODUCTION

Mashups have been making an impact from the informal use of multiple services in a web page to a more formal discussion of emerging issues, trends and tools. Linux Journal [1] had an article in July, 2006 on creating mashups. Jackson and Wang [9] discuss security issues arising from the creation of mashups from multiple sources. In a keynote to the 32nd International Conference on Very Large Databases [10] Jhingran spoke to the "fundamental transformation" affected by mashups and their impact on enterprise architectures. He hypothesized the emergence of a new architecture, "enterprise information mashup fabric", to address newer integration technologies.

That mashups are becoming more mainstream is apparent from the emergence of developer tools for creating them. Wong and Hong [19] described an end user programming tool called Marmite and indicate that its ease of use is demonstrated by how spreadsheet users (and programmers) had little difficulty using the system. Another tool, the Web Mashup Scripting Language (WMSL) also provides end user capability using no more than a web browser to "quickly write mashups that integrate any two, or more, web services on the Web [17]."

Finally, that mashups are an important topic for inclusion in Information Technology curriculum becomes evident by recent articles in the press, notably an article highlighting Google Maps in the New York Times in December, 2006 [16].

At our University, we have tried, specifically in our Information Systems and Computer Science graduate curricula, to incorporate emerging technologies.

## 2. INCORPORATING WEB-BASED APPLICATIONS IN THE CURRICULUM

The computing division of the university offers two Information Technology masters degrees (as well as masters in computer science and Software development). The MS in Internet Technology is a concentration oriented degree in which students can study E-commerce, security or open source development. The MS in Information Systems is also concentration-based and has a foundation that can be waived by students with an undergraduate degree is IS. The MS/IT core overlaps with the MS/IS foundation so that both groups of students are grounded in core IS concepts (data communications, database management and project management) and can share concentrations and electives.

*XML Application Development* is an elective for all graduate students. The course introduces XML and covers web services with XML application development. The major topics of the course include:

- Basics of XML
- Cascading Style Sheets
- Extensible Stylesheet Language Transform (XSLT)
- Extensible Formatting Objects (XSL-FO)
- Document Type Definition (DTD)
- XML Schemas
- Basics of Web services
- Introduction to Ajax

Prerequisites for the course include knowledge of HTML and JavaScript. Students learn how to create a mashup with a specific example and then are asked to create their own. Since this is an emerging area, there is no textbook for the course. Several texts are suggested for reference [2,8,15].

## 3. CREATING A MASHUP

A mashup is a Web application that is developed using two or more existing applications or services. In the mashup created in this paper, web services provided by Google Maps and Yahoo Traffic Web are used. Google Maps provides a highly interactive mapping interface that can be used to display various kinds of data on maps. The controls provided on maps can be used to make navigation very easy. Furthermore, it is easy to customize and incorporate them into your Web pages. Most users are familiar with these controls when using Google Maps and have seen them incorporated into different commercial websites.

There are several steps in creating the mashup:

1. Create an HTML page using the Google Maps web service.
2. Use JavaScript within the webpage to input an address from the user
3. Convert the address to longitude and latitude using the Google Maps web service
4. Call a server side CGI script to send the location to the Yahoo traffic service
5. The Perl CGI program sends the address to Yahoo traffic service
6. The CGI script receives an XML traffic report on the location and returns the result to the originating client
7. The client JavaScript processes the XML document, creates markers with the traffic reports at the locations and places them on the map.

## 4. USING THE GOOGLE MAPS API

Students are first asked to go through the Google API documentation [4]. Google Maps currently supports Firefox 0.8+, Mozilla 1.4+, IE 5.5+, Safari 1.2+, Netscape 7.1+, and Opera 7+. The API uses JavaScript to interact with the Google Maps services.

The first step is to get a Google Maps API key [6]. Unlike APIs in Java, which has a huge number of functions, Google Maps API is a very small collection. Students can learn the whole API in a couple of hours, but like learning any API one concentrates on a small subset in the beginning and adds more as you become more experienced.

## 5. THE BASIC INTERFACE

Google provides an excellent documentation for the APIs [4,5]. The following HTML statements are the building blocks of the Google side of the web-based application.

The BODY tag

```
<body onload="load()"
onunload="GUnload()">
   <div id="Mymap"
style="width:w; height:h"></div>
</body>
```

When this page is loaded, the JavaScript function *load()* will be executed. The DIV tag is important. This is a space holder for the map. Map will be loaded here in the rectangle whose width and height (two numbers in pixels) are provided. Notice the DIV tag has a label.
Example:

```
<body onload="load()"
onunload="GUnload()">
   <div id="myMap"
style="width:500;
height:600"></div>
</body>
```

The map will load in a rectangle that is 500 x 600 pixels. The name of DIV tag is *myMap*.
class GMap2 (they all start with G – Google).
This is the central class in the API. Everything else is auxiliary. We instantiate class GMap2 in order to create a map. An instance of GMap2 represents a single map on the page. A typical way of creating an instance of a map in JavaScript:

```
var mapObject = new
GMap2(document.getElementById("
DIVTagLabel"));
```

The argument *DIVTagLabel* is the name of the DIV tag in BODY of the document. *mapObject* is a variable representing the map object.

Example:

```
var map = new
GMap2(document.getElementById("
myMap"));
```

This creates a map object, called *map*.

## 6. SETTING THE MAP

After creating a map instance, you set it on the screen using the method, setCenter():

```
mapObject.setCenter(new
GLatLng(latitude,logitude), zoom);
```

Here *latitude* and *longitude* are the latitude and longitude of the location and *zoom* is a number indicating the zoom value. Zoom values are from 0 through 17. A zoom level of 0 shows the entire earth and zoom level 17 zooms to the street level. The map will be centered in the sense that the specified latitude and longitude will be at the center of the displayed map.

Example:

```
map.setCenter(new GLatLng(43.083,
-79.95),13)
```

The map will be loaded in the space holder provided by the DIV tag, with center of the rectangle showing the point 43.083 latitude and -79.95 longitude (Niagara Falls), with zoom value 13.
Note:

1. GLatLan() is a constructor of the GLatLan class, which inputs longitude and latitude of a point on earth and returns a point object.
2. Actual loading of the map is done when you invoke setCenter() function, which is done in the load() function:

```
function load() {
      var map = new
GMap2(document.getElemen
tById("myMap"));
        map.setCenter(new
GLatLng(40.77, -74.00),
13);
  }
```

3. You have to enter the latitude and longitude in decimal format.

There are many Web sites which give you latitude and longitude of a location [3,11]. Figure 1 (see appendix) illustrates a basic map of New York City created from the Google API illustrated above.

## 7. ENHANCING THE MAP

Controls are added with the *addControl()* method. In this case, we add the built-in *GSmallMapControl* and *GMapTypeControl* controls, which let us pan/zoom the map and switch between Map and Satellite modes, respectively. Figure 2 shows the controls.

**Figure 1. Map Navigation Controls**

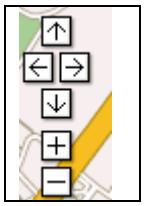

**Figure 2. Map Marker**

Markers. Markers at points specified by their latitude and longitude, using GMarker class shown in Figure 3.

Click event handling. The action of a user clicking on a map is called a click event. When a click event occurs, we can make our JavaScript execute a function.

```
GEvent.addListener(markerObject,
"click", function(){
        function body
});
```

If the click event is associated with a marker, a click on a marker will cause the body of the function to be executed.

One can modify the characteristics of marker icons. According to Google, "The most difficult part of creating a marker is specifying the icon, which is complex because of the number of different images that make up a single icon in the Maps API."

## 8. CONVERTING AN ADDRESS TO LATITUDE AND LONGITUDE: GEOCODING

Geocoding is the process of converting an address to longitude and latitude so it can be placed on a map. Google Maps APIs require latitude and longitude of a point. Version 2 of the Google API provides geocoding capability with a geocoder class, GClientGeocoder.

The basic idea is to create an instance of GClientGeocoder class and then call the getLatLng() method:

```
geocoder = new GClientGeocoder();
geocoder.getLatLng().
```

The getLatLng basically takes two arguments: the first is the address for which you are looking for latitude and longitude, the second is a function which inputs the LatLng point generated (this is done automatically). Once we have the latitude and longitude of a location, we can map it.

## 9. YAHOO TRAFFIC WEB SERVICE

Yahoo Traffic Web service [18] inputs the address of a location and returns, as an XML document, the traffic information around the location. You need a Yahoo ID to use their services.

We need an HTML document and server side CGI script. The HTML page inputs the address and sends the data to the CGI script. The CGI script (written in Perl) receives the data, sends a request to the Yahoo Traffic Web service, and sends back to the client browser the data sent by Yahoo Web service. This data will be in XML format as illustrated in Figure 4.

## 10. A MASHUP OF GOOGLE MAPS AND YAHOO TRAFFIC SERVICE

With the building blocks in place one can now develop an application which combines Google Maps and Yahoo Traffic service. As indicated earlier, we build the map and geocode the address with Google Maps services, and then retrieve the traffic information from the Yahoo Traffic services. The last step is to build and annotate the markers with the Google Maps API discussed above.

Figure 5 displays a Google map embedded with traffic information as given by Yahoo Traffic service in user specified location.

The red markers indicate severe problem and yellow markers routine problems. When you click a marker it describes the current status at that location (see Figure 6).

## 11. CONCLUSION

Because of the increasing number of web services available (including the Web sites

referenced in this paper), the number of hybrid applications that can be created is limitless. We believe that students should be exposed to the latest technology. The value of this technology (and technique) is in the ability to rapidly build a variety of types of applications. Understanding of how to create applications, program API's, and deploy innovative websites is as asset to potential employers as well as the knowledge base of an information technology professional. Finally, we believe that the ability to create complex applications from existing web services is one way of making the information systems and computer science curriculum current and more exciting.

## 12. REFERENCES:


At the forge: Creating mashups. Linux Journal, Volume 2006, Issue 147 (July 2006), p10.

Daconta, M. D., Saginich, A., XML Development with Java 2, Sams, 2000.

Degrees, Minutes, Seconds and Decimal Degrees Latitude/Longitude Conversions, Federal Communications Commission, Available: http://www.fcc.gov/mb/audio/bickel/DDDMMSS-decimal.html.

Google maps API. Google maps, Available: http://www.google.com/apis/maps/.

Google maps documentation. Google Maps, Available: http://www.google.com/apis/maps/documentation/reference.html#GMap2.

Google maps signup. Google maps, Availabile: http://www.google.com/apis/maps/signup.html.

Gorgone, J. T., Gray, P., Stohr, E. A., Valacich, J. S., Wigand, R. T., (2006), MSIS 2006: Model curriculum and guidelines for graduate degree programs in information systems. Communications of the Association for Information Systems (17), 1-56.

Harold, E. R., Processing XML with Java: A Guide to SAX, DOM, JDOM, JAXP, and TrAX, Addison-Wesley Professional, 2002.

Jackson, C., Wang, H. (2007), Subspace: secure cross-domain communication for web mashups. Proceedings of the 16th international conference on World Wide, 611-620, Banff, Alberta, Canada.

Jhingran, Anant, Keynote Address (2006). Proceedings of the 32nd international conference on Very large data bases, 2006, Seoul, Korea.

Look-up Latitude and Longitude – USA . Bahai Computer and Communication Association (BCCA) Available: http://www.bcca.org/misc/qiblih/latlong_us.html.

Mashup (web application hybrid,) Wikipedia, the free enclopedia, Available: http://en.wikipedia.org/wiki/Mashup_(web_application_hybrid).

Mashups, APIs and the Web as Platform. Programmable Web , Available: http://www.programmableweb.com/.

Mashups, APIs and the Web as Platform. Programmable Web , Available: http://www.programmableweb.com/howto

McLaughlin, B., Java & XML, 2nd Edition: Solutions to Real-World Problems, O'Reilly Media, Inc., 2001.

O'Connel, P. M. , The New York Times, December 11, 2006, Online, Available: http://www.nytimes.com/2005/12/11/magazine/11ideas1-13.html.

Sabbouh, H., Higginson, J., Semy, S., Gagne, D., (2007), Web mashup scripting language. Proceedings of the 16th international conference on World Wide Web, 1305-1306, Banff, Alberta, 2007.

Traffic Web Services - Traffic REST API. Yahoo Developer network, Available: http://developer.yahoo.com/traffic/rest/V1/index.html.

Wong, J., Hong, J., (2007), Making mashups with marmite: towards end-user programming for the web. Proceedings of the SIGCHI conference on Human factors in computing systems 1435-1444, San Jose, CA.

## APPENDIX

**Figure 3. Generated Google Map**


**Figure 4. Returned XML from Yahoo Traffic Service**

```
<?xml version="1.0" ?>
- <ResultSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="urn:yahoo:maps" xsi:schemaLocation="urn:yahoo:maps
http://api.local.yahoo.com/MapsService/V1/TrafficDataResponse.xsd">
  <LastUpdateDate>1151982911</LastUpdateDate>
- <Result type="incident">
  <Title>Lane closed, on RT-119 at I-287</Title>
  <Description>road maintenance operations starting 9:00 AM, 07 05 06
until 3:00 PM, 07 05 06 Comment: Right lane closed.</Description>
  <Latitude>41.064024</Latitude>
  <Longitude>-73.855820</Longitude>
  <Direction>N/A</Direction>
  <Severity>2</Severity>
  <ReportDate>1152104400</ReportDate>
  <UpdateDate>1151731970</UpdateDate>
  <EndDate>1152126000</EndDate>
  </Result>
```

**Figure 5. Google Maps and Yahoo Trafic**

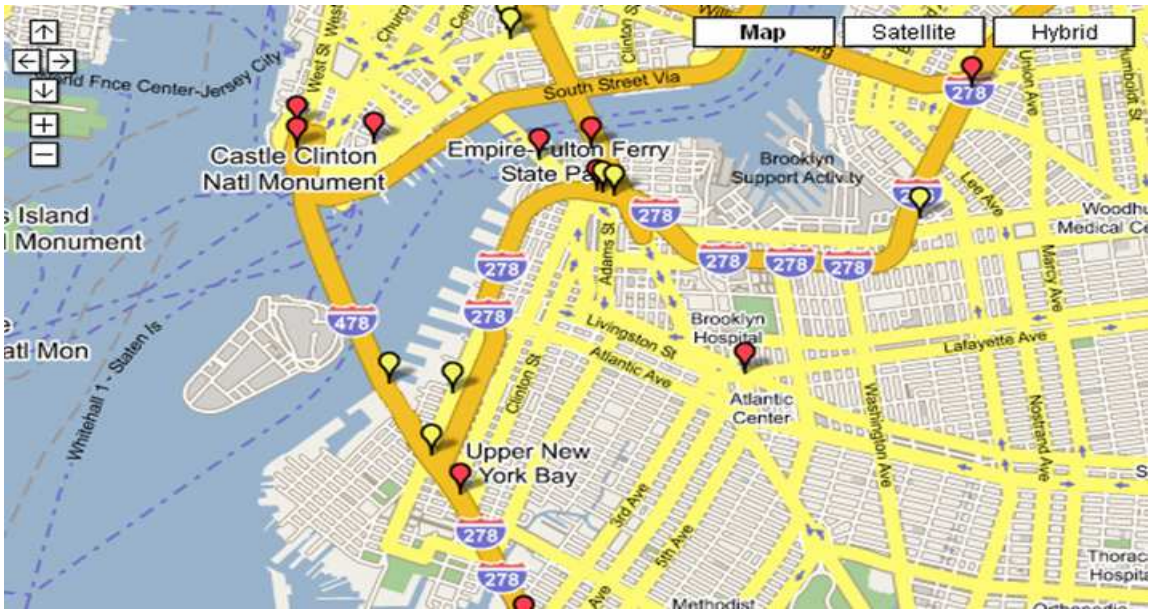


**Figure 6. Annotated Caption**


**Table 1. Other examples developed by students.Other examples developed by students:**

| | | |
|---|---|---|
| Tourist places | James Rennard | http://vulcan.seidenberg.pace.edu/~nmurthy/Jim/TravelPlans.html |
| Indian Restaurants | Ancey Verghese | http://vulcan.seidenberg.pace.edu/~nmurthy/Ancey/IndianRestaurants.html |
| Golf Courses | Julia Lizardi | http://vulcan.seidenberg.pace.edu/~nmurthy/Julia/GetGolfCourses.html |