Web 2.0 Mapping Mashup Applications for Academic Universities–Comparative Case Study (Secure Google Maps API versus Microsoft Virtual Earth API)

Ms. B.Padmaja Rani* M.Upendra Kumar**

**Associate Professor CSE JNTUCEH Hyderabad A.P India*

***Associate Professor CSE MGIT Hyderabad A.P India*

padmaja_jntuh@yahoo.co.in,uppi_shravani@rediffmail.com

Web Mapping is the process of designing, implementing, generating and delivering maps on the World Wide Web. A common challenge for many web developers is the creation of web applications that capture and describe geographical information. As a result of Web 2.0 technologies such as Asynchronous JavaScript and XML (Ajax) web-based mapping applications with rich contents can be developed with better interactivity, functionality, accuracy and usability. These applications (for example Academic Universities Web Mapping) provide users with maps to search and browse geographic information. The integration of information on maps as a layer is known as Mashups. The Value of a Mashup lies in better user interface for that data or in combining data from a number of sources in an interesting way. In this paper we want to propose Web mapping strategies for Academic Universities, using robust Secure Google Maps API version 2, and Microsoft Virtual Earth API with a comparative study of API's involving tests on browser compatibility, performance, accuracy, functionality, usability, geocoding and development support etc. We explain how web-mapping works, Google Maps API motivation for such academic application, with our Microsoft Virtual Earth implementation of our Web mapping application of MGIT. We conclude with Map Abstraction Layer which improves interoperability of API's integrating development methodology. Finally issues pertaining to securing Google Maps API will be discussed.

Keywords – Web mapping, Web GIS (Geographic Information System), Google Maps API version 2, Microsoft Virtual Earth, Map Abstraction Layer, Web 2.0 Mashups.

1. Introduction

Web Mapping is the process of designing, implementing, generating and delivering maps on the World Wide Web.[8] It provides maps for users to search and browse spatial information, such as location and routes. Web cartography deals with usage of web maps, the evaluation and optimization of techniques and workflows, the usability of web maps, social aspects etc. Due to the change in technologies, with high bandwidth Internet connections and advanced web development techniques, nowadays geographic data can be provided and transferred across Internet at a low cost, making it possible for everyone to integrate and display a map in a website. Web GIS (Geographic Information System) emphasizes on analysis, processing of project specific geodata and exploratory aspects. Web maps are increasingly gaining analytical capabilities along with acting as presentation media for Web GIS. A special case of web maps are mobile maps, displayed on mobile computing devices, such as mobile phones, smart phones, PDA's, GPS and other devices. The maps on these devices are displayed by a mobile web browser or web user agent, are called mobile web maps which can display context and location sensitive information, points of interest etc.

The use of web as a dissemination medium of maps can be regarded as a major advancement in cartography and opens new opportunities, such as real-time maps, cheaper dissemination, more frequent and cheaper updates of data and software, personalized map content, distributed data sources and sharing of geographic information. Important research issues of web mapping are interactivity, usability, security, reliability etc. Classification web maps include static map, dynamic map, view-only map, interactive map etc. Common Web map functionality includes Viewing modes (Road map mode, Satellite/ aerial mode, Hybrid mode etc.), Navigation (Zooming and Panning), Controls, Overlays, Markers, Information Windows, Polylines, polygons and circles, Driving directions, Geocoding (the process by which information is associated with particular geographic points in the world often identified by degrees in latitude and longitude), Points of interest etc. Web-based

mapping usability involves sophisticated Graphical User Interface (GUI) with the guidelines like: terminology should be clear and unambiguous and designers should avoid jargon; Professional designers should be used to improve the graphic design of the site. The design should enhance and complement the text and the maps, focusing the user's attention on the content; A meaningful legend should be presented as port of the default view of the map so that the map is self explanatory; A locator or context map, that shows where the map is being viewed in relation to a larger geographic data should be provided; Buttons should have text or icons with tool tips showing the name and describing the purpose or action. There should be large enough for users to accurately identify the text or image and to click with the mouse; Help should be provided and there must be a range of appropriate error messages.

1.1. Web Mapping Technology

To create or implement a web-based map, several techniques, any programming environment, programming language and server side framework, can be used. In any case of technology choice, both server and client side technologies have to be used to be able to provide and display dynamic and interactive maps.

1.1.1 Ajax (Asynchronous JavaScript and XML)

Ajax allows web applications to be loaded and updated through several micro requests instead of just one large macro request, which is a traditional model for web applications. Ajax is rather a collection of powerful technologies: standards-based presentation using XHTML and CSS; dynamic display and interaction using the Document Object Model (DOM); data interchange and manipulation using XML and XSLT; asynchronous data retrieval using XMLHttpRequest; and JavaScript binding everything together. [11, 13]. Below Fig. 1. Illustrates Ajax-based web Application model.

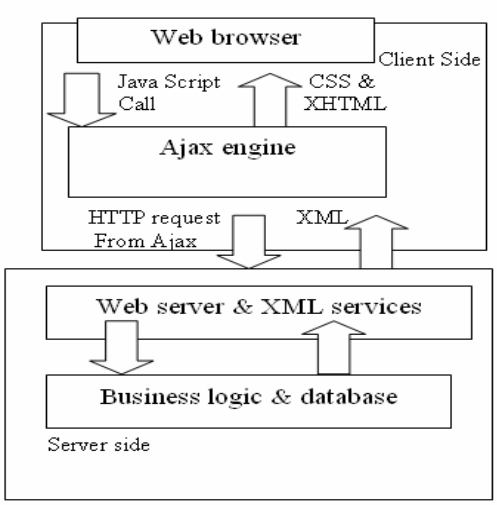


Fig 1 Ajax based web application development.

1.1.2 Web-based maps

When an integrated map is displayed in a user's browser window three parts (a 3-tier application framework) are involved: the user's browser, the web server hosting the site with the map, and the API system. A map system consists of several sets of servers, one set hosting static files, such as JavaScript's and images, one set used to calculate routes and other geographical information and two sets of servers hosting the road map and satellite tile images. On the client side Dynamic HTML (DHTML) and Ajax are used to display the map in the browser window. DHTML is a combination of using the JavaScript event model together with CSS positioning. That is what creates the possibility to drag objects, in this case the map, displayed in web pages. The map is broken up into a grip of images called tiles, which are absolutely positioned square formed images, normally 256 x 256 pixels, displaying the different parts of the map. When the map is dragged new images are fetched from the tile servers as soon as the area is visible and the no longer visible tiles are removed. This removing and adding process creates the maps infinite scrolling effect. [1, 2]. Consider Fig. 2 which depicts Google Maps Tiles.

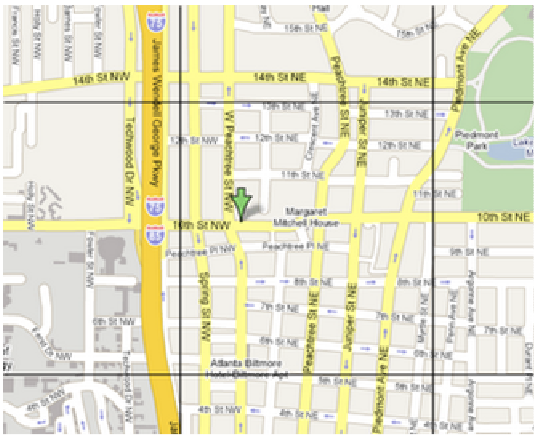


Fig 2: Google maps with tiles outlined.

1.1.3 Geographical data

The geographical data used and displayed by the different web map APIs is in general coming from the same data vendors.[12] The majority of the API's use a combination of data from companies as NAVTEQ and TeleAtlas for the road maps. The satellite/aerial imagery is coming from companies like DigitalGlobe which is photographing the Earths surface with their QuickBird satellite. The satellite makes fifteen orbits around the Earth per day and photographs strips which measure 16.5 by 330 kilometers. The average resolution is 60 square centimeters per pixel but because of the huge competition for the camera most of the planet has only been photographed with a low resolution. In addition to the QuickBird satellite, low resolution imagery is also captured by others satellites as the Landsat-7, which has been taking images of the entire planet at a resolution of 15 meters. After the images have been taken they are sent to ground stations for post-processing. The differences in photographic angle are corrected and the images are resampled so that their pixels will be aligned with a latitude-longitude grid. Many areas of high interest, as cities, are also photographed by aircraft to get a higher resolution; clearly visible in the resulting photos are car sunroofs, lampposts, and even people.

1.2 Features of Web 2.0

The important feature of Web 2.0 over traditional Web 1.0 includes, [5] Creators: Thin line between contributors, developers, and users. In many cases, users themselves contribute and collaborate; User interface: The user interface can be more adaptive, customized and often like that of a desktop application; Content Serving: Using the semantic web (Web 3.0) markup that describes data specifically; Reading content: Using syndication such as RSS/Atom feeds; User experience: Rich; Applications: XML Web Services, example www.ebay.com ; Navigation: Customized and let's the user navigate web pages; Presentation: XHTML, Ajax and CSS. The major building blocks of Web 2.0 include CSS, XML, Web Services; Web 2.0 tools include: Ruby on Rails, Turbo gears, and ASP.NET 3.0/3.5.

2. Overview of important features of Google Maps API version2 [1,2,3,14]

The functionality of robust Google Maps API version 2 supports three Viewing modes: Map view, Satellite view, and Hybrid view. It supports map controls in three sizes map-type control, scale control, overview map, and local search control. Its Overlays include Markers (or Pushpins, optional with a custom icon), InfoWindows associated with the marker and map. It supports Ploylines and polygons to display routes or a polygon highlighting a region, on top of the map. It supports development of custom controls by extending GControl class. It has Marker Manager Utility to provide solution to slow rendering of map and visual cluttering, if the map contains number of markers. It supports Route and driving directions with textual directions and current traffic situation using Mobile GPS receivers. It supports the KML (Keyhole Markup Language and GeoRSS data formats using GeoXML overlay. KML features include Place markers as GMarkers, Icons with a size of 32 x 32 px; Descriptive HTML displayed inside infowindows, Polygons / Polylines, Styles for Polylines/Polygons, Ground Overlays without rotation/opacity. It supports addition of event listeners for mouse movement and keystrokes. It supports the languages: Japanese, French, German, Italian, Spanish, Catalan, Basque, Dutch, and Galician. The major limitation of Google Maps API is that they work only within a browser; thereby embedding a Google Map application in a Windows (or other) application is not possible.

Our example motivation of Academic University Google Map application came from the below figure 3. [15].

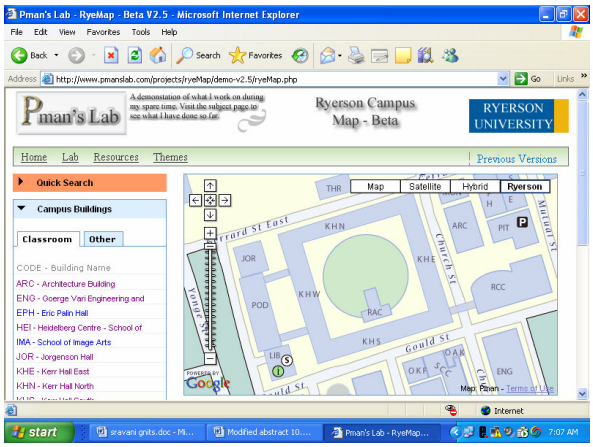


Fig.3. Example of University Google Map application.[15]

3. Overview of important features of Microsoft Virtual Earth Control API [16]

The functionality provided by Microsoft's Virtual Earth Map Control API includes Viewing modes like Road, Aerial, Hybrid, Birds eye view and 3D view. Navigation control includes Dashboard control which comes in three sizes: Normal, Small, and tiny sizes. Controls include Find control and Minimap (overview map). Overlays / shapes include Pushpins with info boxes, Polylines and polygons, Shape layers, Messages, Route and driving directions, GeoRSS, VE Collection layer, and events.

The following plates snapshots depict our MGIT Virtual Earth implementations Fig. 4 depicts pushpins on MGIT across India. Fig. 5 depicts pushpins customized over all blocks of MGIT. Fig. 6 depicts Route map from mehdipatnam to MGIT. Finally Fig. 7 and 8. depicts Route Map across road.

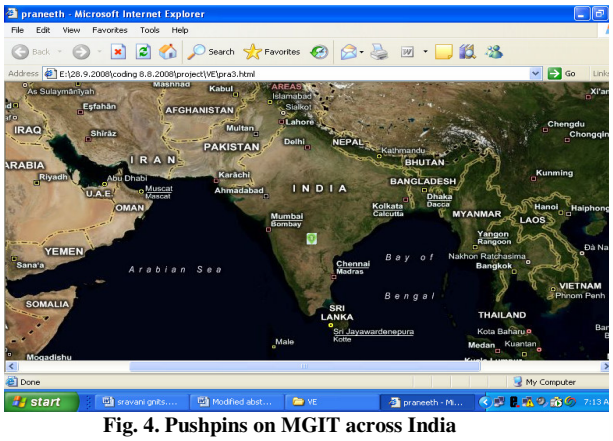


Fig. 4. Pushpins on MGIT across India

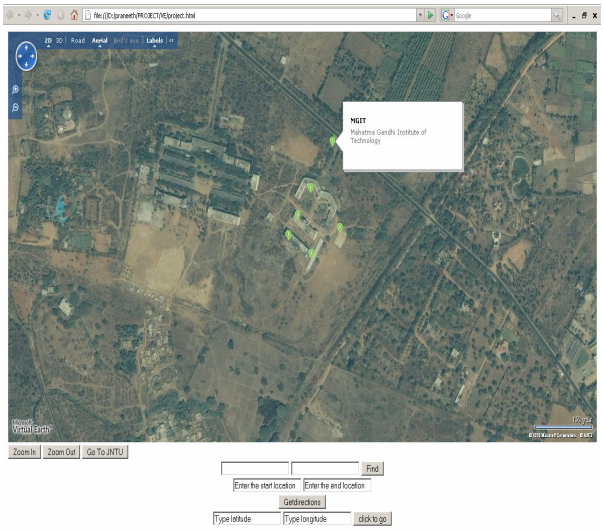


Fig. 5. Pushpins customized over all blocks of MGIT

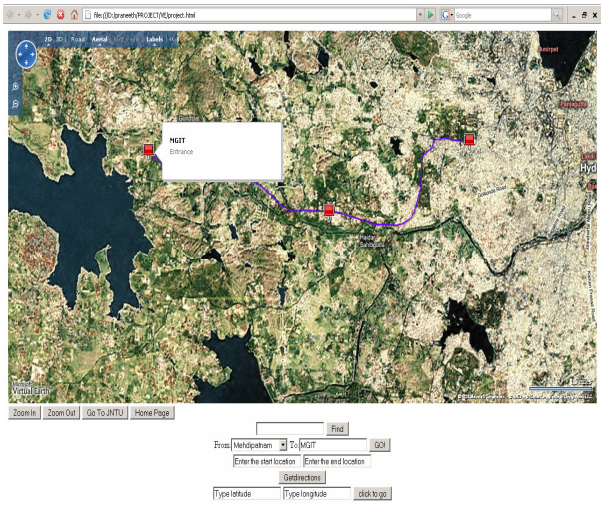


Fig. 6. Route map from Mehdipatnam to MGIT

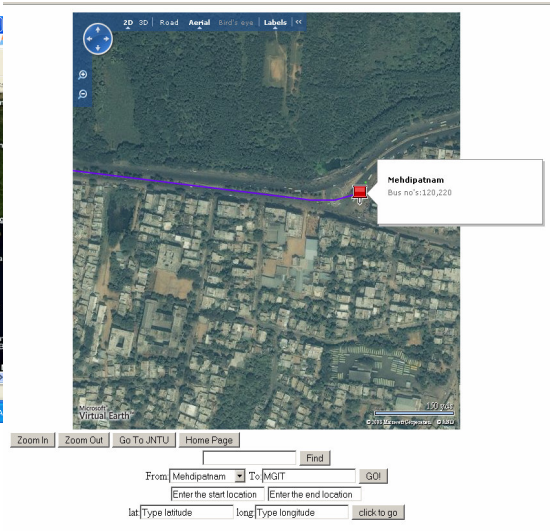


Fig 7. Route map across road.

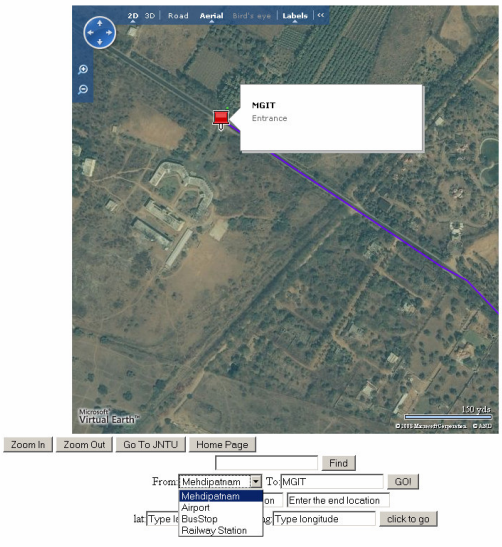


Fig. 8. Road map route (continued)

4. API's Comparative Tests

4.1 Geocoder:

Both Google Maps and Virtual Earth provide geocoding functionality for translation of postal addresses into geographical coordinates. Addresses can be sent either as unparsed texts containing all address information or as multiple address parts such as: street, zip code, city etc. Both API's support unparsed addresses only.

4.1.1 Spherical law of cosines

The distance calculation between two points on a spheres surface, using geographic coordinates, can be done in several ways: Haversine formula, the Spherical laws of cosines and the Vincenty formula. When geocoding tests were performed to measure the geocoder's accuracy the spherical laws of cosines were performed to calculate the distances between points. The formula is defined as:

$D = \arccos(\sin(lat_1).\sin(lat_2) + \cos(lat_1).\cos(lat_2).\cos(long_2 - long_1)).R$

Where $(lat_1, long_1)$ is one point's coordinates, $(lat_2, long_2)$ the other points coordinates, d is the distance and R the spheres radius which in this case was the Earth's average radius, 6374 km. latitude, or lat, is the angle between any point on the earth's surface and the plane of the Equator. Each pole is 90 degrees: +90 degrees at the North Pole and -90 degrees at the South Pole and in-between the equator is placed at the 0-degree parallel of latitude longitude, or lang, is the angle east or west between any point on the earth's surface and the plane of an arbitrary north-south line between the two geographic poles. The line passing through Greenwich, UK, is the international zero-longitude reference line, the Prime Meridian. The Spherical law of cosines is less accurate than the haversine formula, but according to the company Movable Type Ltd, create of a JavaScript implementation of the Haversine and Spherical law of cosines formulas, the numeric precision of JavaScript is so good, 15 significant figures using IEEE 754 floating-point numbers, and that the simple Spherical law of cosines formula gives well-conditioned results down to distances as small as around 1 meter.

4.2 File Load test.

To test the amount of data download when a web-map based map is viewed, the same area was displayed with the road map mode, at the same zoom level and with the same map size. The downloaded files were recorded to see sizes and hosting servers and below table shows the results, the amount of data downloaded. All file sizes are in kilobytes (KB) and Google Maps and Virtual Earth provide their API JavaScript files with compression. Table 1 shows Map API load test results.

Table1. Map API load test results

[KB]	Google Maps	Virtual Earth
JavaScript size	73	146
Tile image size	435	434
Total size	516	606
Tile Servers	4	4

4.3 Browser compatibility

Table 2 shows Map API browser compatibility. .

Table2. API browser compatibility according to map API service providers

Browser	Google Maps	Virtual Earth
Internet Explorer	6.0+	5.5+
Fire fox	0.8+	0.9+
Mozilla	1.4+	-
Netscape	7.1+	-
Opera	8.02+	-
Safari	1.2.4+	-

Table 3 shows Browser compatibility test results.

Table 3. Browser compatibility test results. Working browsers are marked with X

Browser	Google Maps	Virtual Earth
Internet Explorer	X	X
Firefox	X	X
Netscape	X	X
Opera	X	-
Safari	X	-

4.4 API comparison

4.4.1 Viewing modes

Table 4 shows API Viewing Modes.

Table 4. API viewing mode. Supported nodes are marked with X.

Viewing Mode	Google Maps	Virtual Earth
Map/road	X	X
Satellite/Aerial	X	X
Hybrid	X	X
Birds eye	-	X
3D	-	X

4.4.2 Controls (Google Maps and Microsoft Virtual Earth)

Table 5 and Table 6 show Available controls with Google Maps and Microsoft Virtual Earth.

Table 5. Available controls with Google Maps

Control	Functionality
Map control in three sizes	Pan, return to start location and zoom
Map type control	Change map viewing mode
Scale control	Scale indicator
Overview map control	Small overview of the map
Local search control	Location search control(Google Ajax search API)
Custom controls	Yes, possibility to add HTML elements to map

Table 6. MS Virtual Earth controls

Control	Functionality
Dashboard in three sizes	Zoom and change viewing mode
Scale	Fixed scale indicator, not removable
Find control	Loction search control. Where and What search
Mini map	Small overview of the map
3D map control	Pan, rotate and tile the map in 3D mode
Custom controls	Yes, possibility to add HTML elements to map

4.4.3 API provided overlays

Table 7 shows API provided overlays.

Table 7. API provided overlays

Overlay	Google Maps	Virtual Earth
Pushpin/marker	Yes	Yes
Info Window/box	Yes	Yes
Detached info window/box	Yes	-
Tabbed Info Window/box	Yes	-
Message	-	Yes
Polyline	Yes	Yes
Polygon	Yes	Yes
Content Grouping	Marker Manager	Content Layer

4.4.4 Driving Directions

Table 8 shows API driving direction functionality.

Table 8. API driving direction functionality

Feature	Google Maps	Virtual Earth
Multiple Point	Yes	No, only two point direction
Textual descriptions	Collection of plain text directions or preformatted HTML	Collection of plain text directions
Options	-	Shortest or quickest route
Route Information	Distance and duration	Distance and Duration
Traffic information	Current traffic situation in some supported cities	-
Localization	9 Languages	1 Language

4.4.5 Customizability provided by API's

Table 9 shows customizability provided by API's.

Table 9. Customizability provided by API's

Element	Google Maps	Virtual Earth
Pushpin / Marker	Custom icons, changeable at any time	Fully customizable pushpins with HTML or icon image. Changeable at any time
Info window/box	Customizable with PdMarker extension	Customizable with CSS
Controls	Customizable with extensions	-

4.4.6 GeoXML Overlays and layers provided by API's

Table 10 shows GeoXML overlays and layers provided by API's.

Table 10. GeoXML Overlays and layers provided by API's

Data Format	Google Maps	Virtual Earth
GeoRSS	Yes	Yes
KML	Yes	-
VE collection	-	Yes

4.4.7 Usability

Table 11 shows API usability evaluation results.

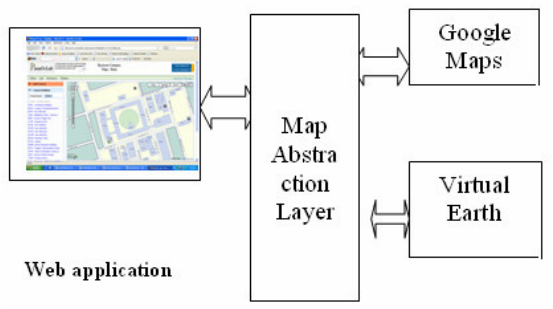
Table 11. API usability evaluation results

Guide line	Google Maps	Virtual Earth
Draggable Map	Yes	Yes
Grouped pan buttons	Yes	No pan buttons
Pan buttons grouped according to directions	Yes	No pan buttons
Tool tips	Localized tool tips for all buttons	English tool tips for buttons tool tips without text
Default zoom	Original Center	Original center
Overview map	Yes, fully intractable	Yes, fully intractable

5. Map Abstraction Layer: [5]

Fig 9 depicts JavaScript abstraction layer in between web applications and map API's.

Fig.9, Java Script abstraction layer in-between web applications and map API's



To eliminate too much dependability on application development API's, the need for a Map Abstraction layer is felt. This layer is placed between the web application and the Map API. It provides a common interface to all included map APIs at the same time and there fore the map used could easily be switched with a small work effort. Most of the methods implemented in the layer will be forwarding method calls from the application to the Map API's, creating a common interface, however since the API's are constructed differently, with different methods and different object types, some common objects have to be created. Latitude – longitude object has to be created for coordinates as well as objects for locations, markers, polylines and polygons. In addition to this some common objects have to be created, such as routes, parts and steps, to provide driving directions for all API's.

6. Securing Google Maps [6]

An enforceable solid security policy should serve as the foundation of any security effort. Without a policy, our safe guards could be insufficient or unenforceable. Google Attack Libraries are broken down into three main components: Socket initialization and establishment; Google Query requesting; and retrieving a Google query response. The steps involved in Psuedocoding include: Initializing socket; Send Google Request; Retrieve Google Response; Scrap Google Responses, and Return total hits. The Python language proved the most useful and efficient for creating Automated Google Query code. Its OOP style, easily accessible regular expression engine, and indexing methods made it easy to create, send, retrieve and scrape Google Information. The C# from Microsoft .NET library is the most extendable language implementation of our Google libraries because it can be merged into any program that's compatible with Microsoft Visual Studio .NET. Conducting Google Vulnerability tests scans is one of the easiest tasks that have hit the information security industry in the past few years. The key to automating such a task is the looping constructs that wrap around the library implementation. We can implement loopin constructs to automate searching and information retrieval for numerous purposes. Nearly all vulnerability scans utilizes the allinurl: advanced searching flag to search for strings stored within the Google cache.

7. Conclusions

In this paper, we proposed the technological issues of working on Web Mapping pertaining to Web 2.0 mashups. We are motivated that every academic university should have such an application which provides such geographic data. We compared the API's Google Maps and Microsoft Virtual Earth, about their strengths and Weaknesses. We provided the need for Map Abstraction: Layer. Finally issues pertaining to securing Google Web Maps from hackers are discussed.

8. References

[1]. Christopher C.Miller, " A Beast in the Field: The Google Maps Mashups as GIS/2", Cartographica, The International Journal for Geographic Information and Geo Visualization, Volume 41, issue 3, pp. 187 – 199, 2005.
[2] "Is Google Maps GIS?" (http://highearthorbit.com), GIS India Journal, July 2008, page 8 – 10.
[3] Dionysius G. Synodinos, "Web Maps with the Google Map API", Dr.Dobbs Journal, December 2007 pp 36 - 41.
[4] Developer IQ Magazine Cover Story,"Mashups", VOL 7 No. 9, September 2007.
[5] Laszlo Zentai, "Application of Web 2.0 in Cartographic education: Is it time for Cartography 2.0?" 2006.
[6] Johnny Long, "Google Hacking for penetration testers", SYNGRESS, 2005
[7] Richard Kendall Et. Al., "Development of a Weather Forecasting Code: A Case Study" IEEE Software, August 2008.
[8] Zng-Ren Peng, Ming-Hsiang Tsou, "INTERNET GIS Distributed Geographic Services for the Internet and Wireless Networks", Wiley, 2003.
[9] Ravindranath Jampani et. Al," Overlaying Multiple Maps Efficiently", International Institute of information Technology, Hyderabad 2001.
[10] Zao Liu et.al, "Implementing a Caching and Tiling Map Server: a Web 2.0 Case Study", 2006.
[11] Young Liu, Et. Al., "Leveraging Web 2.0 technologies in a cyber Environmental Research", OGF-19, Semantic Web 2.0 and Grid Workshop, January 29, 2007, NC, USA.
[12] A.M.Chandra, S.K.Ghosh, "Remote Sensing and Geographical Information Systems", Narosa Publications, 2006.
[13] Michael Purvis et. Al., "Beginning Google Maps Applications with PHP and Ajax", APress, 2006
[14] Manlai You, et. Al., "A Usability evaluation of Web Zoom and Pan functions, International Journal of Design, Vol. 1, No. 1, 2007.
[15]http://www.pmanslab.com/projects/ryeMap/demo-v2.5/ryeMap.php
[16] Jon Arking, "Professional Windows Live Programming", Wrox Publication, 2008.