MapBeing: An Architecture for Manipulating and Publishing Vector Data in Web based Geographic Information System

Md. Moshiur Rahman
BSSE 0330

A Thesis
Submitted to the Bachelor of Information Technology Program Office
of the Institute of Information Technology, University of Dhaka
in Partial Fulfillment of the
Requirements for the Degree

BACHELOR IN SCIENCE AND SOFTWARE ENGINEERING

Institute of Information Technology
University of Dhaka
DHAKA, BANGLADESH

MapBeing: An Architecture for Manipulating and Publishing Vector Data in Web based Geographic Information System

Md. Moshiur Rahman

Approved:

Signature *Date*

_____ _____

Supervisor: Dr. Kazi Muheymin-Us-Sakib

Abstract

Web Geographic Information Systems (WebGIS) have been introduced for fulfilling the growing need of Geographic Information Systems (GIS) in the Internet. WebGIS can also be used to manipulate the published data according to the user's need. However, publishing of large volume interactive vector data might degrade the performance of WebGIS, due to upload/download and rendering time. In this paper, a vector data manipulating and publishing architecture named MapBeing is proposed which provides features to manipulate and publish huge volumes of data on maps without degrading performance. The proposed architecture consists of four core layers named User End, the Request Handler, Service and Data Provider. The User End layer is constructed using an open source JavaScript library called OpenLayers which shows vector data on the interactive map presented on the web. The Request Handler layer processes data provisioning based on user interaction. GeoServer is used in the Service layer for map creation and modification and PostGIS, a Database for storing spatial data, is used as the Data Provider layer. The experiment shows that along with the ability to manipulate and publish vector data, the architecture also achieves significant amount of lower usage in user bandwidth.

Acknowledgments

First and foremost, I would like to thank Dr. Kazi Muheymin-Us-Sakib for his support and guidance during the thesis compilation. He has been relentless in his efforts to bring the best out of me.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Geographic Information System (GIS) is a collection of computer hardware, software and data for managing, analyzing and displaying all forms of geographic information. The vector data format is one of the spatial data types to store geographic data such as points, lines, polylines and polygons. For the proper usage of this data, a ubiquitous technology is needed for accessing this data from anytime and anywhere such as smartphones or desktops. In order to address those aforementioned issues, WebGIS has been introduced for publishing and interacting vector data using web technologies. However, such publishing of large volume geographic data on the map is really challenging in the web platform as excessive data transfer will be required which will degrade the system performance. WebGIS also provides several value-added services to its users such as modifying and sharing of vector data [1]. It has been widely recognized that future developments in GIS will be based on web-based GIS having access and analysis of geospatial data on the web. While WebGIS is gaining momentum in mainstream consumer usage, the core manipulation of spatial data such as create, edit and delete should be considered with utmost attention.

1.1 Motivation in WebGIS

For a highly interactive WebGIS based map, users often need to modify vector data on the web for creating and updating geographical models such rivers, cities, towns, roads, restaurants, hotels, hospitals, ATM booths and other infrastructures. For this purpose, the server sends whole vector data to client sides which might be desktop browsers, smartphones or even smartwears. Those client sides are then

responsible for the rendering, displaying and modifying of vector data obtained from server side data providers. The main problem in such mechanism is the limited processing power and storage of aforementioned client devices. There is a network bandwidth cost as well for downloading the whole vector data file every time from server to client devices which is infeasible often in real life scenario as well.

1.2 Issues in WebGIS

In the last few decades, authors have made some remarkable contributions to the GIS development in this context. TerraFly [2] and GeoHosting [3] are instances of those which focused on the publishing and analyzing of vector data. S. Puri et. al. proposed a distributed algorithm for large scale vector data overlay processing [4]. A WebGIS framework for Vector Data sharing has also been proposed by F. Yin et. al. [1]. However, none of those contributions considered manipulation of vector data. Moreover, existing technologies for communication between server and client side involves only tile images which are the representation of those huge vector data residing on the server side. Vector data will be rendered here on the fly as per the requests from clients with the required parameters. Hence, it is really challenging for the users to edit or delete geospatial shapes on the client devices since they are provided only partial images of the total vector data. In order to address these challenges, the research proposed MapBeing which can modify vector data on the map as well as publish with open source server side GeoServer, client side OpenLayers and data storage PostGIS [5]. MapBeing is also fully capable of publishing and handling interactivity of Environmental Systems Research Institute (ESRI) provided shape files[6] on the map without considering the volume. Spatial data services are developed to control the interactions between the browser and the spatial data. Finally, based on the proposed architecture and the services, a spatial

data management and visualization system has been designed and implemented using the latest Open Source projects. PostgreSQL and PostGIS are used to manage and store GIS spatial data and its metadata. GeoServer performs as the map server, an ASP.NET web server is used as the MapBeing server, OpenLayers is used as client side mapping and interaction with vector data.

1.3 Contribution

Experiments have been performed for the use case demonstration of the proposed architecture using Open Source vector data as well as comparison with one of the architectures which sends all the vector data to the client side. The Architecture has been implemented as a web application in the .NET managed environment. There are several modes of operation in the application from which user can select the required one from the toolbar. In the MapService Mode which is the default mode, OpenLayers shows tile images obtained from the server no matter how large the vector data volume is. In the Edit mode, the image tile to be edited, can be selected and manipulated. The selected image tiles vector data are obtained from the server and after editing, it can successfully save the edited vector data. User can also draw and save data as vector format in Drawing Mode. As the application does not get the whole vector data at a time, the response time is better than conventional applications and bandwidth cost is also reduced. In comparison with a production level GIS solution which consumes all the vector data from the server side, a significant amount of the bandwidth cost is reduced as a lower amount of vector data have been communicated between the client and the server.

1.4 Research Questions

As stated in the previous section, the nature of MapBeing raises the following research question :

- Are the existing WebGIS architectures suitable for supporting large amount data manipulation on the web?

To be more specific:

- Are the existing WebGIS solutions enough for publishing vector data as well as manipulating those data on the web?
- If the existing frameworks are not supportive enough, then what modifications are necessary to make it more usable?

The main aim of this research is to answer the questions mentioned above and thus providing a solution for performing manipulation of vector data on the web.

Chapter 2

Background

A geographic information system (GIS) is a computer system for capturing, storing, checking, and displaying data related to positions on Earths surface. GIS can show many different kinds of data on one map. This enables people to more easily see, analyze, and understand patterns and relationships. Science and technology has played quite a major role in providing information through advanced research techniques. In this regard, spatial analysis is one way through which science is providing information. Spacial analysis with GIS basically involves studying the geographical information systems of the world.

2.1 Evolution of the GIS

"The early days of GIS were very lonely. No-one knew what it meant. My work has certainly been missionary work of the hardest kind." — Roger Tomlinson, Father of GIS 1968

"The application of GIS is limited only by the imagination of those who use it" — Jack Dangermond, Esri

"GIS is a form of digital mapping technology. Kind of like Google Earth, but better." —Arnold Schwarzenegger at the Government Technology Conferences 2008 Conference on Californias Future.

A GIS or Geographic Information System combines software, hardware and information for analyzing, managing and capturing all geographical forms to display as reference information. The acronym GIS is also used for Geospatial Information Studies that refers to the academic study or discipline that works along with geographical information system. In the laymans terms, GIS means the incorpo-

ration of statistical analysis, cartography and the database technology. It has the power to relate information from many different sources and incorporate it into one database software.

The history of GIS development started primarily in 1832 spearheaded by the French geographer Charles Picquet when he applied spatial analysis in epidemiology. Starting from there, John Snow used the method possibly meant as the earliest use of geographical method when he depicted a cholera outbreak in 1854. The advancements in technology that has been recorded as the history of GIS development made the present GIS as it is today. In the 20th century, the printing of geographic locations were already made possible. However the images are not yet considered to be vital as there are no databases to link to them.

Canada then developed the first and really operational GIS. It was called Canada Geographical Information System or CGIS and was being used in 1960 to save, manipulate and study the data gathered for Canada Land Inventory. It has been an improvement from the computer mapping software because it gives researchers the ability to scan, overlay and measure geographical places. The CGIS lasted up to the 90s but was never marketed as a product.

By early 1980s a team of different companies surfaced as GIS software sellers as they have successfully combined the CGIS features with their upgraded development into it. By the end of the 20th century, the growth of GIS platform and systems has been rapidly spread that many users are already viewing GIS data using the internet. Right now, there are available customized platforms of GIS that performs many different tasks making mapping applications and geospatial data are already available in the internet.

There are a lot of available articles over the web that also provides more about the history of GIS development and advancements for anyone to make reference with.

2.2 Evalution of WebGIS

Geographic information systems (GIS), as a way to manage, map, and analyze geographic information, and the World Wide Web, as a way to make this information accessible to as many people as possible, are natural allies. Therefore, the term Web GIS has been around for years as a category for products and services. For example, the Urban and Regional Information Systems Association (URISA), used to give an annual Best-Web GIS award intended to increase awareness of Internet/Intranet mapping services that enable publishing and sharing of geographic information internally within an organization or externally with the public.

Recently, the concept of Web GIS has been getting yet additional attention, because most of the latest technological advances in GIS have been in the area of on-line services and because access to spatial data as well as advanced mapping and spatial analysis over the Internet are becoming more common. In his opening remarks at the annual ESRI International User Conference in San Diego in early August, attended by 14,500 people, the companys president, Jack Dangermond, referred repeatedly to Web GIS, which was also the subject of several very popular technical sessions in the following days.

Web GIS refers to more complex and sophisticated geographic data analysis activities than the more common term Web mapping (which includes the Microsoft Virtual Earth and Google Earth platforms). It also points to the future of GIS, which, after a long predominance of desktop applications, is now transitioning back to a client-server architecture whether the client be on a desktop or a mobile application.

This transition is a result of two things. First, what economists would call supply push: the availability of larger, faster, and more reliable servers and networks. Second, what they would call demand pull, consisting of several elements:

- Increased demand for access to a myriad data services such as satellite

imagery, aerial digital orthophotos, real-time tracking, etc.,

- Increasing need for off-site storage of huge datasets including high-resolution imagery and LiDAR 3D point clouds downloaded from data services or collected in-house, and - End-users increased expectations for access to their data anytime and anywhere. The only thing that is slowing down this transition is lingering security concerns about storing expensive and often mission-critical data on other companies machines.

2.3 Data handled by WebGIS

There are various kinds of statistical data that can be collected and thus the need for analysis in order to have meaning. This is the sole reason as to why there is the need for special analysis with GIS. This is also to ensure that most people are able to understand the statistical findings as they are simplified. This is more so because thus is not a field that most people are conversant with. The geospatial information system uses space time index for all the information collected in the database. This is one special technique that is employed in this field. Accuracy of the statistical analysis depends on quantitative analysis of the data collected as this tends to bring key issues to focus.

2.3.1 GeoJson

GeoJSON [7] is a format for encoding a variety of geographic data structures. GeoJSON supports the following geometry types: Point, LineString, Polygon, MultiPoint, MultiLineString, and MultiPolygon. Lists of geometries are represented by a GeometryCollection. Geometries with additional properties are Feature objects. And lists of features are represented by a FeatureCollection.

2.3.2 ESRI's Shape File

A shapefile [6]stores nontopological geometry and attribute information for the spatial features in a data set. The geometry for a feature is stored as a shape comprising a set of vector coordinates. The shapefile format is a popular geospatial vector data format for geographic information system (GIS) software. It is developed and regulated by Esri as a (mostly) open specification for data interoperability among Esri and other GIS software products.The shapefile format can spatially describe vector features: points, lines, and polygons, representing, for example, water wells, rivers, and lakes. Each item usually has attributes that describe it, such as name or temperature.

2.3.3 Raster Data Format

Raster [8] data models incorporate the use of a grid-cell data structure where the geographic area is divided into cells identified by row and column. This data structure is commonly called raster. While the term raster implies a regularly spaced grid other tessellated data structures do exist in grid based GIS systems. In particular, the quadtree data structure has found some acceptance as an alternative raster data model. The size of cells in a tessellated data structure is selected on the basis of the data accuracy and the resolution needed by the user. There is no explicit coding of geographic coordinates required since that is implicit in the layout of the cells. A raster data structure is in fact a matrix where any coordinate can be quickly calculated if the origin point is known, and the size of the grid cells is known. Since grid-cells can be handled as two-dimensional arrays in computer encoding many analytical operations are easy to program. This makes tessellated data structures a popular choice for many GIS software. Topology is not a relevant concept with tessellated structures since adjacency and connectivity are implicit

in the location of a particular cell in the data matrix.

2.4 Existing Mapping Services

Web GIS/mapping products and services include:

2.4.1 ALOV Map

A free, portable Java application for on-line publishing of vector and raster maps and interactive viewing on Web browsers; it supports complex rendering architecture and allows users to work with multiple layers, thematic maps, hyperlinked features, and attribute data.

2.4.2 AutoDesks MapGuide 6.3

AutoDesks MapGuide 6.3 a Web mapping client/server technology with Java client/applet that helps users develop, manage, and distribute GIS and design applications on-line, offers point-of-activity access to interactive maps, designs, and data, and allows users to serve spatial and attribute data from GIS, CAD, and other spatial databases.

2.4.3 Google Earth and Google Maps

Google Earth a desktop virtual globe application that combines satellite imagery, geographic data, and Googles search capabilities and Google Maps a fast and easy way to access maps and to geographically enable Web sites.

2.5 Summary

The evolution of cloud computing is discussed in this chapter. Moreover, the true notion of cloud computing XaaS is also introduced. Services that include but not limited to XaaS such as IaaS, PaaS, SaaS, HUaaS and STaaS are reviewed one by one. The inter-relationships between them are discussed too. Finally an introduction to TFP is given.

Chapter 3

Literature Review

The GIS development has drawn attention of the researchers over the last few decades for its growing usage in business, research and education etc. Day by day, more efficient and high volume data rendering capable GIS is needed. Hence, GIS researches focuses on efficiently vector data publishing, analysing, sharing and also modification. In this section, the most notable work on GIS has been highlighted.

3.1 Generic Architecture

Many definitions for the term GIS have been proposed in the literature, each considering the functionality of the system from different perspectives. All of them focus on three different aspects: GIS is a collection of computer tools to perform geographic analysis and simulations,
GIS is supported by a set of data structures and algorithms to represent, retrieve and manipulate geographic information ,
GIS is a utility that helps people make decisions in tasks related to geography.

3.2 GeoHosting

GeoHosting [3] proposed a WebGIS architecture called GeoHosting as a Spatial Data Infrastructure (SDI). The main objective of GeoHosting is to offer services supporting the creation of a spatial data sharing system with possibility to publish data for any user having access to the Web. As it is developed focusing on spatial data sharing compatibility, it lacks the T of modification of vector data.

3.3 TeraFly

TeraFly [2] is a high performance web-based GIS. It uses Internally Distributed Multithreading (IDMT) method to achieve high performance and semantic R-tree to search an object on both spatial and semantic information. It gives a high speed heterogeneous data publishing architecture but it provides give direction to modify (Edit, Delete and Update) large vector data efficiently.

3.4 Parallel and Distributed for Overlay Processing

R. puri et. al. [4] developed a Parallel and Distributed algorithm for large scale Polygon Overlay processing with the help of MapReduce [9]. It can find the result of a query consisting of two vector overlay layers with large amount of vector data and it is implemented on General Purpose Graphics Processing Unit. Although this algorithm can find result by processing two vector overlay but cannot edit data of those overlays.

3.5 Vector Data sharing

F. Yin et. al. [1] proposed a WebGIS framework for Vector Data sharing based on open source projects. It focuses on practical implementation of large vector data sharing and interoperability. The authors have divided this application into the application layer, service layer, function layer and storage layer using open Source software such as PostgreSQL, GeoServer, OpenLayers and TileCache etc. As it is developed to facilitate user for sharing large scale data, it does not consider the modification of vector data on the map which is very important for an interactive mapping system.

3.6 Spatial Data Infrastructures (SDIs)

A Spatial Data Infrastructures (SDIs) has been implemented in this paper by Steiniger et. al. [10]. The proposed architecture is based on Free and Open Source Software (FOSS) [11] so that users can buy a full SDI with limited cost. The SDI is developed with Web Map Servers,Web-GIS Servers for data processing, spatial DBMS for storage, Registry/catalogue and metadata software. In this architecture, a complete GIS is proposed but interaction between vector data and users on the map is not considered.

3.7 GIS in Water Resource Management

An Web based GIS System has been developed the Water Resource Management [12]. It used open source software to build a WebGIS. It is capable of publishing, storing vector data as well as raster data. XMLParser is used here while sending and receiving data format for communication between server and client. This architecture was developed and applied to spatial data management and utilized that data for problem analysis and solving such as monitoring natural disasters, solving water insufficiency in agricultural areas etc. The main part of this work was used at GIS in water resource management field. Hence this architecture does not help users to modify or to create vector data from map.

3.8 Summary

From the literature, it has been found that manipulation of large vector data in WebGIS is still not fully matured. Currently, the browser itself does the collaboration between GIS and vector data. However, browser cannot handle large amounts of data because of limited resources and low processing power. Thus,

new architecture for vector data publishing in GIS is required.

Chapter 4

MapBeing: An Architecture for Manipulating and Publishing Vector Data in Web based Geographic Information System

Studying existing architectures on webGIS endures that the architectures do not directly address vector data manipulation on the web. Different architecture tried to solve different part of webGIS challenges but modification, creation, deletion of vector shapes on the web is also challenge. Thus a new architecture is required to support these features. Keeping these factors in mind, a web based architecture named MapBeing is proposed here.

4.1 Overview

A new vector data manipulating and publishing architecture for GIS solutions is proposed in this paper. GIS data set is the spatial representation of real-world features such as roads, buildings and terrains. The interaction, modification, deletion, presentation, analysis and capture of vector data are all done by GIS. For manipulating vector data, MapBeing sends just the data of specific shape which is requested from the client side.

During the framework design phase, priority was given on the modification of large amount of vector data on the map and raster data which are not possible to be handled by the browser. The architecture also focuses on the time issue which is a great problem in client side data rendering. Besides, the manipulation of vector data is unavailable in current GIS views. This facility is as well provided in this new framework architecture

The top level view of the framework architecture is shown in Figure 4.1. The architecture is divided in three sections and one database -

- Web Browser
- MapBeing Server
- GeoServer
- PostGIS Database

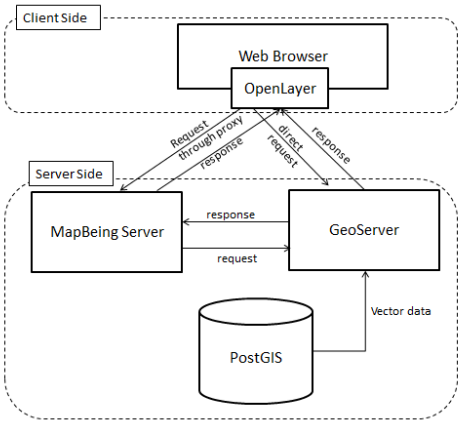

Figure 4.1: Top Level View of MapBeing

The web browser has the task to response to the user interaction on the map. This task is accomplished by using a library, OpenLayers. OpenLayers receives the user requests and sends it to the MapBeing server. Again the response of the MapBeing server is shown to the user by it. The MapBeing server is used here for the manipulation purpose of the vector data. However, the OpenLayers directly requests to GeoServer to display the data. It acts as a transparent layer on the map which makes a canvas on it and draws the shapes responded by the server. The MapBeing server section is responsible for connecting the OpenLayers with the GeoServer. As a web server, it has the conventional task to provide web content to the browser through the Internet; and processing server scripts and sending an appropriate response to a client request. However, the main responsibility of the

MapBeing server here is to process user manipulation requests and sending those to the GeoServer. Different services of GeoServer are used for different purposes which will be mentioned later in this chapter. Some of the services can be used directly from browser while others require the MapBeing server to be in the middle of some processing work. It takes user clicked position in the map images through the OpenLayers, gets the vector data information of position manipulation and finally sends request to GeoServer.

The GeoServer [13] is the most important section of the framework. It provides three services -

- Web Feature Service (WFS)
- Web Map Service (WMS)
- Web Coverage Service (WCS)

WFS provides an interface for making requests to get geographical features. While basic WFS allows only querying and retrieval of features, a transactional WFS (WFS-T) [13] allows creation, deletion, and updating of those. It processes http request of clients and executes appropriate operations to serve them. WMS is a standard protocol that serves geo-referenced map images over the Internet. These map images can be generated by a map server using data from a GIS database. Although it usually serves the map in bitmap format, vector graphics can also be served in SVG or WebCGM format. Digital geospatial information representing space time-varying phenomena is retrieved by WCS. It provides access to coverage data in forms. It allows clients to choose portions of a server's information rather than the providing whole.

The Web Browser directly communicates with the GeoServer through WMS service. The map images got from this service is displayed on the browser. As these are the only images, these cannot be manipulated directly like vector data. This is where the MapBeing server is needed to process the clicked image position to

get the specific vector data and asks for manipulation. This is done by WFS-T service of the GeoServer.

The PostGIS database is used for vector data storage. It provides data to GeoServer which offers its services. The user gets the services through browser. This solution limits the browser tasks to manipulate vector data that automatically enhances client-side performance.

4.2 Architecture of MapBeing

The component stack of the proposed framework is represented in Figure 4.2. It is separated in three layers:

- User End Layer
- Request Handler Layer
- Service Layer
- Data Provider

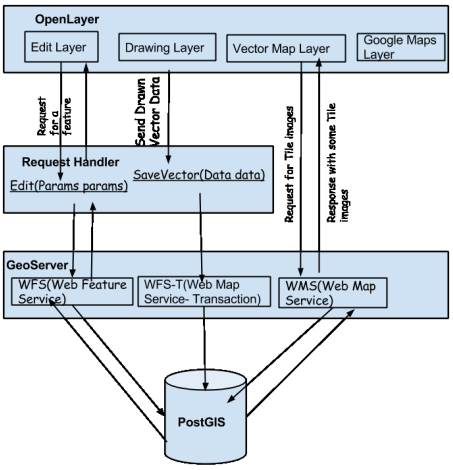

Figure 4.2: Architecture of MapBeing

Each layer has different responsibilities. The User End Layer which is actually the OpenLayers, interacts with the users viewing locations in map. It has the responsibility to take user commands and show respected change in the map. The Request Handler Layer processes user demands and requests Service Layer for required services appropriately. The Service layer provides the three previously mentioned Geo services to the User End directly or through Request Handler and thus can be called as GeoServer Layer. The Data Provider layer stores vector data and supplies it to the GeoServer on demand.

The first layer or the OpenLayers consists of four components- Edit Layer Component, Drawing Layer Component, Vector Map Component and Google Map Component. OpenLayers is a transparent layer on the map where the locations are drawn. It is a canvas with points, lines, rectangles and various other shapes representing the vector data locations. Google Map is the map provided by Google which is used here for the background of the vector data locations in the browser. The Vector Map Component requests directly to the GeoServer with map b-box, zoom-level and layer information as parameters. As a response, it gets some tile images and places those on the map. The Edit and Drawing components are used for the map data manipulation. Both of these components contact with Request Handler. The Edit Layer is responsible for editing map features on OpenLayers. And the Drawing Layer is responsible for saving the manipulated information.

Request Handler layer is the web server layer which is in a sense, the middle layer between OpenLayers and GeoServer Layer. It has the task to do the manipulation processes of vector data. It consists with two components- Edit and Save Vector. The Edit Component takes user interaction information on the OpenLayers tile images as input and provides the vector data information about that area. It also makes all the manipulation of the data. The Save Vector Component takes the drawn data information about users and saves it using GeoServer Service.

The third layer is the GeoServer layer. It provides the WFS, WMS and WCS ser-

vices to the User End and Request Handler Layer. And it also offers WFS-T which is an advanced WFS service. For manipulation task WFS and WFS-T services are used. These services are accessed through Request Handler layer. The viewing is done by using WMS Service. It is directly approached by the OpenLayers.

The fourth layer of MapBeing is Data Provider layer. It is the storage layer and the warehouse of the framework. The vector data which can also be termed as geographical data is stored here. The database used for this storage is PostGIS Database.

In principle, User End Layer does the user interactivity with vector data, Request Handler receives requests and sends responses to User End Layer, Service Layer responsible for rendering, manipulating of vector data and finally Data Provider stores user data. This forth layered architecture achieved the minimization of data bandwidth cost of manipulation of vector data in web based GIS.

Chapter 5

Experimental Results

5.1 Environment Setup

In this section, our WebGIS system architecture is discussed in more detail. The vector layer overlay techniques are also discussed. The editing vector layer is also briefly discussed here

The proposed architecture is based on Free and Open Source Software. MapBeing used OpenLayers for client side, GeoServer as web server and PostGIS for storing spatial data. MapBeing also have a server to take XMLHTTP request from OpenLayers. OpenLayers requests to GeoServer with map's view-port and zoom-level as request parameters for tile images. Again, GeoServer request to PostGIS for vector data according to request parameters and creates tile images of those vector data on the fly. Then the tile images are sent to OpenLayers and OpenLayers renders those images on the map-overlay. When OpenLayers need a specific shape's data on mouse click then it requests through MapBeing server to GeoServer. The detailed technical description is given below.

OpenLayers is an open source object oriented mapping library based on HTML5 canvas element. It also has capability to draw with SVG (Scalable Vector Graphics). It is divided into three parts as Map Layers, Edit Layer and Drawing Layer.

Map Layers

Map Layers is responsible for displaying tile images on above base layer and base layer can be Google Maps, Open Street Maps or Microsoft Virtual Earth. More than one layer can be added at a time such as Road Map, Administrative Map, River Map, and Rails Way Map of a country. For displaying base map, OpenLayers uses with available API of map provider and displaying maps of vector data OpenLayers request GeoServer for tile images.

Edit Layer

When user clicks on the tile images to edit a shape, edit layer request GeoServer with required parameters via MapBeing server for a specific shape's vector data. By analysing parameters GeoServer can uniquely identify a shape and sends vector data of that shape. Now by edit layer this shape data can be edited, modify its properties or any other modification user wants to. After editing, edit layer sends edited data to GeoServer via MapBeing server then GeoServer edit this data in PostGIS.

Drawing layer

Drawing Layer is responsible for creating new map with just sketching. Lines, polylines, polygons, points can be drawn by Drawing Layer. After drawing, drawing layer sends drawn data to GeoServer via MapBeing server and GeoServer stores this to PostGIS by creating new shape file.

MapBeing Server

MapBeing uses asp.net web application as MapBeing server. OpenLayers sends edited data or drawn data to MapBeing server. The MapBeing server receives those data from client-side and after pre-proccessing it sends data to GeoServer. Then GeoServer modifies those data and sends to PostGIS. With WFS , GeoServer sends data to MapBeing server as user request from client side with OpenLayers. After edited received data, GeoServer updates table in PostGIS. With PostGIS-T, GeoServer delete a feature from PostGIS relevant table. MapBeing server has a management module to communicate with GeoServer and OpenLayers. It has a processing module to process in coming request from client-site and added extra information to fit the request parameters for requesting GeoServer. It also has a

data management module by which it maintains data uploading and data authorization.

GeoServer

GeoServer has been customized so that MapBeing can use it as it wants. The modification of catalog.xml made GeoServers data source point to our data storage where MapBeing server uploads users data.

PostGIS

PostGIS is a spatial database extender for PostgreSQL object-relational database. It adds support for geographic objects allowing location queries to be run in SQL. GeoServer just execute some query as MapBeing severs request.
After receiving PostGISs response, GeoServer sends feature data to OpenLayers through MapBeing server.

5.2 Test Result

OpenSource vector data is used to test MapBeing's workability. All the experimental data downloaded from http://www.gadm.org/country or from GeoServer's default provided data. USA population data, comes with GeoServer installation, named 'topp:states' has been used to check modification capability of MapBeing. The Building [14] data is used to show that MapBeing does not consider data size to process them. Windows 7 operating system, Intel Core 2 Duo E8400 @3.00GHz and 2.00 GB RAM is used as experimental platform of MapBeing.
MapBeing can run in several mode of operation. User can select a mode to work with from tool bar. There is a layer chooser tool to choose a layer to work with.

Different types of map such as Google Map, Open Street Map are also available as base layer.

By Default, MapBeing remains in MapService mode displayed at Figure 5.1. When user wants to see a layer, OpenLayers Map Service [15] requests to GeoServer for that layer's tile images. Then GeoServer gives responses to OpenLayers and OpenLayers show tile images where they should be displayed.

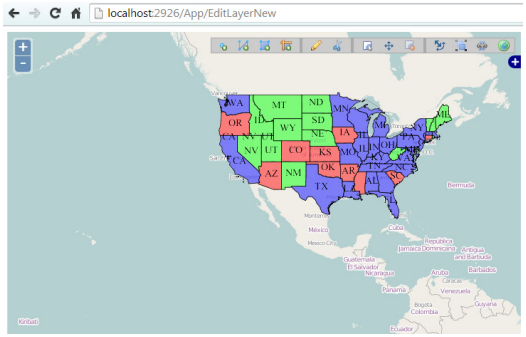

Figure 5.1: Web Map Service Layer

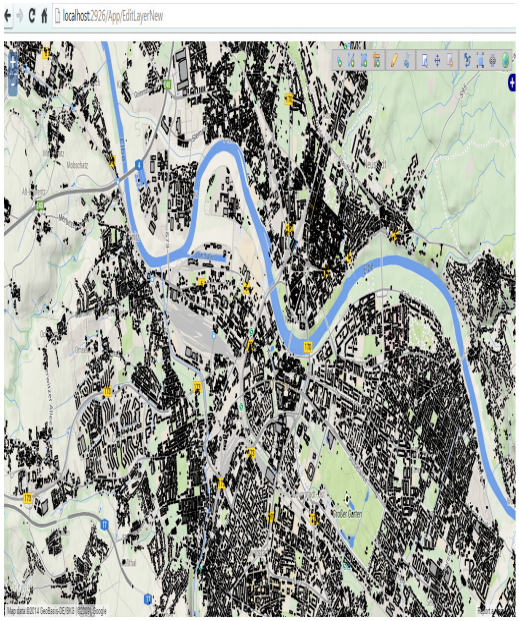

Figure 5.2: Web Map Service Layer (Large Data Set)

The example of large data rendering in the map is shown in Figure 5.2. Without considering volume of vector data MapBeing can capable of display tile images on

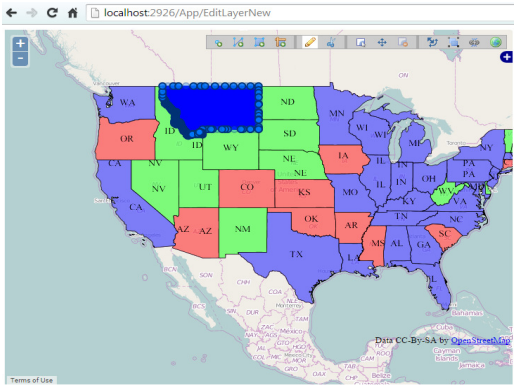

Figure 5.3: Edit Layer

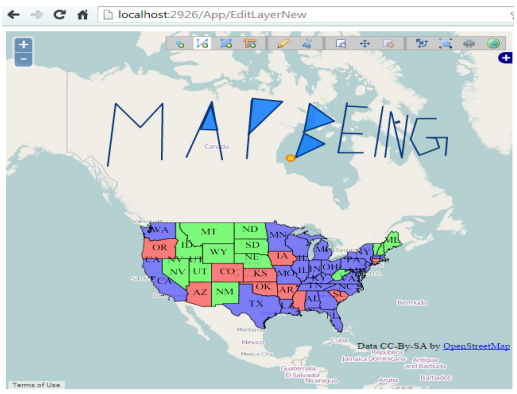

Figure 5.4: Drawing Layer

the map.

Vector data editing mode is shown at Figure 5.3. When User clicks to go to Edit a shape then OpenLayers Edit Service [15] receives click position, Maps's bbox, layer's information and request to MapBeing server. After analysing the request, MapBeing server returns real requested vector data as Json format to OpenLayers. With this data, user can now modify and after modification click save button. Now OpenLayers save strategy send edited data to MapBeing server.

Another value added service of MapBeing is Drawing capability shown at Figure 5.4. User can draw shapes and by using OpenLayers Save Strategy [15] user can save drawn data as vector format. User also can select drawing tools from tools bar situated on above of map window. Lines, Polygons, Point tools are available

ESRIs data name-size	Source of Vector Data	.shp file size(W)	Number of Shapes(N)	Bandwidth in client side rendering	Bandwidth in MapBeing (W/N)
Usa population (0.202 MB)	Geo Server installation	0.183 MB	49	0.183 MB	0.183/49
Virginia Roads (349 MB)	http://download.geofabrik.de/north-america/us/virginia-latest.shp.zip	272 MB	755483	272 MB	272/755483
gis.osm_roads(14.6)	http://www.diva-gis.org/gdata	7.68 MB	50000	7.68 MB	7.68/50000
bangladesh-latest.osm(12.5)	http://download.geofabrik.de/asia/bangladesh.html	10.2 MB	4000	7.5 MB	10.2/4000
India Roads(328 MB)	http://www.geofabrik.de/data/shapefiles.html	240 MB	85516	240 MB	240/855165
Indonesia Roads(126 MB)	http://download.geofabrik.de/asia/indonesia-140908.shp.zip	88 MB	344713	88 MB	88/344713

Table 5.1: Comparison of downloaded data size between MapBeing and client side rendering architecture

to draw.

There is a comparison at Table I on the basis of downloaded amount of data between server side rendering GIS and client side rendering GIS architecture. User can publish, analysis and manipulate vector data on the map by client side rendering GIS. But at the first time of map display, client side rendering GIS downloads whole .shp file from server side to client. On contrary MapBeing downloads just few tile images rendered in the server side. To compare downloaded data size, we defined a Comparison Matrix (CM). From data storage, we know numbers of shape in ESRI's shapefile [16] and size of that shape file:

N = number of shapes in .shp file;

W = size of .shp file in Mega Byte(MB)

CM = W/N.

The larger value of CM means that more data should be downloaded from client side to server side.

As we see in Table I, client side rendering architecture download whole .shp file from server side to client side but MapBeing does not do that. MapBeing download only CM sized vector data as users request. As MapBeing does not consider data size, it performs well even when data size increases.

On the basis of simulation and result analysis discussed above, MapBeing is very efficient in downloading vector data file for manipulation purposes such as create, edit and delete. It downloads only the corresponding shape's vector data at time when user wants to edit instead of downloading the whole vector data from the server. Thus the proposed architecture facilitates both the manipulating and publishing of geospatial data on the web based GIS as well as keeping the data transmission level significantly lower.

After analysing open source sample data set on above, it can be said that server side rendering architecture is more efficient and effective in real-world applications. It is also shown that server side rendering architecture, for example MapBeing, is a more realistic architecture than client side rendering architectures.

5.3 Summary

In MapBeing architecture, vector data publishing and manipulating is very easier and efficient by minimization of data downloading.

Chapter 6

Conclusion

WebGIS has gained popularity because of the persistent availability of geographic information from everywhere, However, a barrier in the mainstream adoption of WebGIS is the transfer of large vector dataset between server and client sides. The unavailability of data manipulation platform in the web based map is another limitation of currently available WebGIS solutions. To address these problems, MapBeing is proposed in this work which facilitates both manipulating and publishing vector data on the web based GIS.

6.1 Discussion

The proposed architecture also handles the problem of large data transfer in map publishing on the web by user interaction based data provisioning techniques rather than providing the whole map-data at a time. The feature toolbar of manipulating the vector data on the map is also presented here. The WMS service of GeoServer is used in this architecture for generating the tile images to be shown on the map by OpenLayers. On the basis of user interaction (click for example) on a position of the map, WFS service is used to get the corresponding vector data. Thus rendering time and data transmission bandwidth are minimized as well. Finally WFS-T is used to save the manipulated vector data on the server side.

6.2 Future Work

The future scope of this research includes providing of a multiple user based map manipulation interface for the same set of vector data. Although, this research only focuses on shape file format, in future, other types of vector data format such as GeoJSON[7], KML[17].

Bibliography

[1] F. Yin and M. Feng, "A webgis framework for vector geospatial data sharing based on open source projects," in *Proc. of 2009 International Symposium on Web Information Systems and Applications (WISA09)*, pp. 124–127, May 2009.

[2] N. Rishe, S.-C. Chen, N. Prabakar, M. A. Weiss, W. Sun, A. Selivonenko, and D. Davis-Chu, "Terrafly: A high-performance web-based digital library system for spatial data access.," in *ICDE Demo Sessions*, pp. 17–19, 2001.

[3] K. Charvat, P. Horak, M. Vlk, J. Cepicky, and S. Kafka, "Geohosting–publish your spatial data yourself," *IST Africa, Kampala May*, 2009.

[4] S. Puri and S. K. Prasad, "Efficient parallel and distributed algorithms for gis polygonal overlay processing," in *Proceedings of the 2013 IEEE 27th International Symposium on Parallel and Distributed Processing Workshops and PhD Forum*, pp. 2238–2241, IEEE Computer Society, 2013.

[5] L. Hsu, "Postgis in action," 2011.

[6] E. ESRI, "Shapefile technical description," *An ESRI White Paper*, 1998.

[7] H. Butler, M. Daly, A. Doyle, S. Gillies, T. Schaub, and C. Schmidt, "The geojson format specification," 2008.

[8] R. J. Hijmans and J. van Etten, "raster: Geographic analysis and modeling with raster data," *R package version*, vol. 1, pp. 9–92, 2012.

[9] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[10] S. Steiniger and A. J. Hunter, "Free and open source gis software for building a spatial data infrastructure," in *Geospatial free and open source software in the 21st century*, pp. 247–261, Springer, 2012.

[11] S. Steiniger and E. Bocher, "An overview on current free and open source desktop gis developments," *International Journal of Geographical Information Science*, vol. 23, no. 10, pp. 1345–1370, 2009.

[12] P. Deeprasertkul and R. Chitradon, "An internet gis system to support the water resource management," *information science*, vol. 1, no. 01, 2012.

[13] J. Deoliveira, "Geoserver: uniting the geoweb and spatial data infrastructures," in *Proceedings of the 10th International Conference for Spatial Data Infrastructure, St. Augustine, Trinidad*, 2008.

[14] "Toulouse: streets, railways, waterways," July 2014.

[15] E. Hazzard, *Openlayers 2.10 beginner's guide*. Packt Publishing Ltd, 2011.

[16] F. LIU, J.-x. ZHANG, and H.-t. LI, "Research and application of shp file format in land over/use map updating [j]," *Science of Surveying and Mapping*, vol. 6, p. 042, 2006.

[17] S. Bacharach, "Ogc approves kml as open standard," 2008.