IMPLEMENTATION OF ONLINE ROUTE FINDER FOR ADDIS ABABA CITY

By

Ayele Admassu

A Project paper submitted to the School of Graduate Studies of Addis Ababa University in partial fulfillment of the requirements for the Degree of Master of Science in Computer Science

June, 2008

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

FACULTY OF INFORMATICS

DEPARTMENT OF COMPUTER SCIENCE

IMPLEMENTATION OF ON LINE ROUTE FINDER FOR ADDIS ABABA CITY

By

Ayele Admassu

Name and Signature of advisor and Examinor:

1. Dr. Dejene Ejigu (Advisor) __________________________
2. Dr. Solomon Atinafu (Examiners) __________________________

Acknowledgement

First of all, I would like to express my sincere gratitude to my advisor Dr. Dejene Ejigu for his guidance, encouragement and advice throughout this project.

I would like to express my appreciation to my family in general; particularly to my wife; Alemshet Bekele, and my father; Admassu Ayele for their relentless support and encouragement in my study. I would like to thank all my classmates and colleagues who have encouraged me throughout the process and read the draft and gave me valuable comments. Thank you Bitsat Tirfe, my mentor and my model boss, this couldn't be realized without your unreserved support.

Last but foremost, I thank God for giving me the wisdom and the strength I need to discharge my duty.

Table of Contents

List of Tables

List of Figures

Acronym

ADF = Application Development Framework
AJAX = Asynchronous JavaScript and XML
API = Application Programming Interface
CD = Compact Disk
DOM = Document Object Model
DVD = Digital Versatile Disc
ESRI = Environmental Systems Research Institute
FAQ = Frequently Asked Questions
GIF = Graphics Interchange Format
GIS = Geographic Information System
IIS = Internet Information Server
JPEG = Joint Photographers Expert Group
KML = Key Markup Language
OGC =Open Geospatial Consortium
PDF = Portable Document Format
PNG = Portable Network Graphics
SLD =Styled Layer Description
SOC = Server Objects Container
SOM = Server Objects Manger
SVG = Scalable Vector Graphics
SWF = Shockwave Flash
TIFF = Tagged Image File Format
URL = Uniform Resource Locator
WMS = Web Map Service
XML = Extensible Markup Language

Abstract

As it applies to any map, web map has very wide spectrum of applicability which includes locating stuff and helping one to figure out how to reach it. Using Internet as a medium, maps can be served on demand to users. Maps that are displayed in web browsers can be grossly divided into two - static maps and dynamic maps. The static ones are composed once and are served with out any change for each and every request made for them. On the other hand, dynamic maps; the focus of this project, heavily interacts with user and are delivered based on ever changing input parameters. Due to this they have to be recomposed, redrawn and be served frequently and hence they are resource intensive.

This project has picked the challenge to deliver online routing advice map service for Addis Ababa City. Shortest or quickest best route analysis is one of the many web mapping service currently available. In that regard, an effort has been exerted to assess the available web mapping technologies that support the mentioned service and tried to deliver a working solution for the local context. From the achievement of the project one can draw a lesson that through merging available technologies user experience can be enhanced. Besides, adopting such kind of techniques has impact on the day to day life of the city residents by enabling them to save time and energy spent in seeking location information by providing map based local information.

Seen from vantage point of the national context such kind of service has greater input for tourism for it can extensively and intensively be used by tourists. The system helps them to have richly informed plan for their stay. Besides, Internet based maps that can be served on demand, can be made dynamic enough to reflect reality and come up with format that the user expects. This in turn increases user's satisfaction. The more is the satisfaction, the more is the chance to get gross revenue form tourism industry.

The core tools and technologies used by the project include ArcGIS Server and ArcGIS Desktop software packages from ESRI and Microsoft's ASP.NET 2.0. The desktop software was used to develop the base mapping service and the server is used to stage the mapping service so that Internet Information (ISS) Server can access it and serve requests from browser. The user interface was developed using ASP.NET 2.0 technology for it has a set of pre built web mapping controls that facilitate development work.

Chapter1

Introduction

1.1 Background

Web mapping is the process of designing, implementing, generating and delivering maps on the World Wide Web. While web mapping primarily deals with technological issues, web cartography additionally studies theoretic aspects: the use of web maps, the evaluation and optimization of techniques and workflows, the usability of web maps, social aspects, and more.

Often the terms web GIS and web mapping are used synonymously, even if they don't mean exactly the same. Web GIS is similar to web mapping but with an emphasis on analysis, processing of project specific geodata[i] and exploratory aspects. In fact, the border between web maps and web GIS is blurry. Web maps are often a presentation media in web GIS and web maps are increasingly gaining analytical capabilities.

Web mapping also implicates many challenges due to technical restrictions (e.g. low display resolution and limited bandwidth), copyright, and security issues, reliability issues and technical complexity.

While the first web maps were primarily static, due to technical restrictions, today's web maps can be fully interactive and integrate multiple media. This means that both web mapping and web cartography also have to deal with interactivity, usability and multimedia issues.

With the help of the Internet and accompanying tools, creating and publishing online interactive maps has become easier and rich with options [1]. For example, a city guide web site can use maps to show the location of restaurants, museums, and art venues. A business can post a map for reaching its offices. The state government can present a map showing average income by area.

[i] *Geodata is a shorter name of Geographic Data. Geographic data is created by manipulating geographic (or spatial) data.*

These days, one of the most popular uses of web mapping is delivering map base local information. Good examples to such kinds of web mapping services include Google Earth, Google Maps, Yahoo Maps, and Windows Live to mention few. These prominent mapping services, which inspired the idea behind this project, deliver very streamlined map based services, which include searching for business and its location, shortest or quickest possible driving route between to points, etc…. Despite their age of launch [2], they are gaining popularity and are attracting wide mass of internet surfers. But, unfortunately to the dismay of many of us, they do not deliver most of their functionality to countries like Ethiopia as they do to any country in North America or East Europe.

Tantalized by the capability of web mapping service and seeing the potential that can be realized with that technology, and seeing the gap that the existing web mapping service giant have in regard to our country context, this project intended to apply the available web mapping technology to local context by taking Addis Ababa as a case. Since web mapping and associated services that can be delivered through it are tremendous in number, this project, in light with available budget and time, has focused in exploring the possibility of realizing web mapping services like driving direction suggestion and searching a location of a business.

1.2 Statement of the Problem

Using web mapping service facilitates daily life of mankind and reduce the cost of obtain information. Tourists, guests, travelers and the likes yearn for local information like location of things and the best route from a point to point as they came into new places. These days the first place to go for getting that kind of information is the Internet. Hence, Web Mapping services that deliver information that can be consumed by the aforementioned groups of people are of paramount importance and plays pivotal roll in providing the basic set of technological solution.

As it can be easily observed, web mapping solution that serve Ethiopian context is not common. There is no web map business location service. Besides there is no web service that advices online users which streets that they can use to travel from a given point to a given point in time.

1.3 Objective

The general and specific objectives of the project are described below:

General Objective

The objective of the project is to develop a web mapping site that provides map based driving route advice and business location searching for Addis Ababa City.

Specific Objectives

1. Develop a Web Site in which users can enter the beginning and the destination information and obtain driving direction map
2. Integration street traffic information dynamically into the system so that users can choose which streets to avoid
3. Enable users to search for a business and see where its located
4. Enable users to enrich the database by allowing them to business location and share them with others
5. Integrate the site with known mapping and business service search providers like Google in order to boost the user experience

1.4 Methodology

In developing the system, first and for most review of works on online mapping services has been done. The literature review is briefly described in chapter 3 of this paper.

The methodology employed to realize the project is object oriented software design and development heuristics. The major milestones and the process steps that were passed through are outlined as follows.

Step 1: Application requirement study

In this step the functional and non functional requirements of the application is thoroughly elicited and documented

Step 2: Application design

In this step the application architecture is determined and components is thoroughly analyzed and documented.

Step 3: Application Development

The first step is to create digital road network map. In order to create it a base map is needed. Fortunate enough, such kind of base map is freely provided by Google and can be obtained form the public Google's map site[i] for free. For the purpose of digitizing and generating the road network digital thematic map, ArcInfo[ii] was used. The road network was analyzed for consistency using ArcGIS Network Analyst[iii] extension. There after the map was posted on ArcGIS Server[iv] to be served up on request. Regarding the user interface development, it was carried out using ASP.NET technology.

Step 4: Testing and Deploying

After the components are implemented unit testing and integration testing were conducted and found satisfactory.

1.5 Justification of the Work

As the advancement in web mapping technology is gaining momentum and as the integration and usage of such services in existing site or new sites is growing for they deliver very localized and visual reach information, it would be an inviting challenge to embrace the technology and explore the possibility of its use in Ethiopian context.

This project has picked the challenge to deliver online routing advice map service; one of the many mapping service, for Addis Ababa City. In that regard, it has tried to assess the available web mapping technologies and tried to deliver a working sample solution for the local context. From the achievement of the project one can draw a lesson that adopting such kind of techniques has impact on the day to day life of the city residents by enabling them to

[i] http://maps.google.com
[ii] *ArcInfoprovides all the tools to build and manage a complete, intelligent GIS including maps, data, metadata, geodatasets, and work flow models (http://www.esri.com/software/arcgis/arcinfo/about/features.html)*
[iii] *Network Analyst enables to solve a variety of problems using geographic networks including shortest path between any number of points. (http://www.esri.com/software/arcview/extentions/networkanalyst)*
[iv] *ArcGIS server is a complete and integrated server-based geographic information system (GIS) that provides spatial data management, visualization spatial analysis services.*

save time and energy spent in seeking location information by providing map based local information.

Seen from vantage point of the national context such kind of service has greater input for tourism for it help them have richly informed plan. Tourist's movement may be restricted since their knowledge for locations and localities is too little.

Maps, if served on demand and if dynamic enough, can reflect reality and can be served in format that the user expects increase users satisfaction. The more the satisfaction, more is the chance to get gross revenue form tourism.

1.6 Scope and Limitations

The project is constrained by the fact that all streets in Addis Ababa City, as it holds true to all other Ethiopian cities, are not properly named. Some are known by local names and others are known by their name given by the responsible authority. For example, the street that spans form Meskal Square to Bole International Air Port is locally known as Bole Road, while its official name is Africa Avenue. Besides, it is very difficult to precisely determine the starting and ending points of a given street. These aforementioned points made it difficult to communicate with users the result of route analysis in common and standardized way. To cope with the problems, the streets were named using a common ground references[i] starting and ending that signifies the starting and ending point of the street.

One of the expected functionality of the system is to deliver optimal driving route information between two points by considering real time ground traffic reality. But this project is constrained by the fact that real time traffic information service is none existent. For circumventing it, the information is abstracted as a web service that delivers XML file format and for the study purpose random values for each street was assigned.

Due to cost and time factors, the project has scoped itself in studying 25km square area for Addis Ababa by centering Aratkilo.

[i] *Like buildings, usual or customary area names, etc…*

Chapter 2

Literature Review

Online mapping has come a long way in the last few years as do many web based technologies. Here follows the literature review that evaluated existing five major contemporary map services and examined their features and performance.

2.1 Google Map

In February of 2005, Google announced the debut of a new service, Google Maps, which changed the face of mapping and cartography on the Web for an overwhelming number of people [3]. Google Maps is a web mapping service that solves the same old problem of online mapping.

Google, in its corporate philosophy [4], states that "The interface is clear and simple" and "Pages load instantly." In addition to these features, Google Maps also offers a number of innovations in web user interfaces. This includes Single search box, integrated local search, draggable maps, traffic maps, terrain maps, satellite imagery, and keyboard short cuts [5].

Google's map and directions service is out in a full release version called Google Local [6]. Google Local rolls the phone book, maps, and driving directions into one big, interactive ball that's handy and just plain fun to use. Google Local employs a split-screen layout: on the right is the interactive map, and on the left is the directory information. With Google Local, one can search on address, types of business, or specific businesses by name, or you can find directions from point A to point B.

As with almost all free software products (and in fact, many paid software products) support options are limited. Still, Google provides a Google Local-specific help page with FAQs and a searchable help section, as well as an in-depth tour of Google Local.

2.2 Yahoo Map

The other major competing online map service provider is Yahoo. Unlike Google, Yahoo has chosen to use Flash over AJAX[i] in building its new service, and have added new features that are not found anywhere else. It's built with the new MacroMedia Flex platform for flash. Yahoo has also done things to reduce the hassle of creating, sharing and printing online maps. [7]

Because Yahoo Maps is a flash application, it avoids the need for page refreshes when changes are made to the page. A simple drag and drop module on the top right lets users move around easily (even easier than the great Google drag method). There are multiple ways to zoom into and out of a map, including using the scroll wheel on the mouse, or the page up/down buttons on the keyboard. The arrow keyboard keys also allow for scrolling off-screen to new areas.

There is an excellent integration with Yahoo Local business. Once a map is pulled up, a quick search, such as "pizza" pulls up all local restaurants. A click shows more information, and options to add it to the driving directions itinerary.

Because Yahoo Maps is built on Flash, the "back button" on the browser still works (a problem with AJAX applications). At any time you can back up into older maps and the URL also rewinds. Yahoo Maps requires the installation of Flash 7 or greater.

2.3 Windows Live Map

Windows Live Local is Microsoft's mapping service that enables to mark locations with pushpins and share customizations with other users. Windows Live Map also integrates with a host of other Windows Live products, as well as with the Outlook e-mail and calendar application [8].

The split-screen layout of Windows Live features an array of tools on the left side and a map with drop-down menus of features on the right side. Keyboard shortcuts make navigation handy. For example, just press the plus or minus keys to zoom in or out, or right-click on a

---

[i] ***A****synchronous* ***Ja****vaScript and* ***X****ML (using and enhancement in JavaScript that allows Web pages to be more interactive and behave like local application. Helps to retrieve small amounts of data from the server without reloading the entire page.*

location to plot directions to or from that spot. Windows Live Map provides a unique, 45-degree-angle "bird's-eye" view that lets circling landmarks from near the ground in any of the four directions.

To find a location, one must enter either business data or address data into the two text fields at the top of the page. But the users must take caution as what to type in where because search result will heavily depend on input. This by itself is a little confusing. Here, Google Maps is more intuitive.

2.4 Ask Map

In the world of online mapping services, Ask Maps and Directions 2.0 is unique for providing multipoint directions within any Web browser, but it's not a standout. Like its competitors, Ask Maps provides satellite imagery so that one can see landscapes and neighborhoods from the sky, but this tool lacks business directories and options for personalization[9].

The uncluttered look and feel of Ask Map pages closely mimics those of Google Maps. The right-hand maps pane enables to choose views of road maps, aerial imagery, or topography, and it offers options for printing, e-mailing, book marking, and linking to chosen maps. The left-hand pane displays step-by-step driving and walking directions, but no business listings. Ask Maps' AJAX technology enables it to work in any Web browser.

Ask Map offers keyboard shortcuts that allow to drag around a map with arrow buttons and zoom in or out with the plus and minus keys. Pressing a number button will take to that numbered step in the driving directions.

Like Google Maps, Ask Maps enables search for a place within a single text field, which unlike the two-field approach of Yahoo and Windows Live. Ask also recognizes three-digit airport codes and allows for some natural-language queries. Unlike Google Maps and the otherwise feature-rich Windows Live Local, Ask provides helpful multipoint driving directions. Once a location is found, however, Ask Maps doesn't connect with business directories to help user find services in the area. On the upside, Ask Maps is currently ad-free.

2.5 Mapquest

Mapquest is one of the best-used Internet mapping services. While Mapquest was once considered stunningly easy and efficient, it hasn't kept up with quickly moving times [10].

It offers maps and directions as well as paid services for access from a car, cell phone, or PDA. However, the Mapquest interface looks dated and isn't as easy or fun as rivals Google Maps, Yahoo Maps, or Windows Live Local. Mapquest now offers advanced route options that allow inputting.

Mapquest's front page offers forms where you can input an address to look up, or put in two addresses for directions. This is a tidy way of working, but Google Map's seems rather effective, which lets you input all your information into one text field using a natural language query.

Chapter 3
Overview of Web Mapping

The use of the web as a dissemination medium for maps can be regarded as a major advancement in cartography and opens many new opportunities, such as realtime maps, cheaper dissemination, more frequent and cheaper updates of data and software, personalized map content, distributed data sources and sharing of geographic information. It also implicates many challenges due to technical restrictions (low display resolution and limited bandwidth, in particular with mobile computing devices, many of which are physically small, and use slow wireless Internet connections), copyright and security issues, reliability issues and technical complexity. While the first web maps were primarily static, due to technical restrictions, today's web maps can be fully interactive and integrate multiple media. This means that both web mapping and web cartography also have to deal with interactivity, usability and multimedia issues [2, 3].

3.1 Types of Web Maps

A first classification of web maps has been made by Kraak [11]. He distinguished *static* and *dynamic* web maps and further distinguished *interactive* and *view only* web maps. However, today in the light of an increased number of different web map types, this classification needs some revision. Today, there are additional possibilities regarding distributed data sources, collaborative maps, personalized maps, etc. Many maps may fall into more than one category. Below, individual web map types are discussed.

3.1.1 *Static Web Map*

Static web pages are *view only* with no animation and interactivity [4]. They are only created once, often manually and infrequently updated. Typical graphics formats for static web maps are PNG, JPEG, GIF, or TIFF for raster files, SVG, PDF or SWF for vector files. Often, these maps are scanned paper maps and had not been designed as screen maps. Paper maps have a much higher resolution and information density than typical computer displays of the same physical size, and might be unreadable when displayed on screens at the wrong resolution.[3]

3.1.2 *Dynamically Created Web Maps*

These maps are created on demand each time the user reloads the webpages, often from dynamic data sources, such as databases. The webserver generates the map using a web map server or self written software [2].

3.1.3 *Distributed Web Maps*

These maps are created from distributed data sources. The WMS protocol offers a standardized method to access maps on other servers. WMS servers can collect these different sources, reproject the map layers, if necessary, and send them back as a combined image containing all requested map layers. One server may offer a topographic base map, while other servers may offer thematic layers [12].

3.1.4 *Animated Web Maps*

Animated Maps show changes in the map over time by animating one of the graphical or temporal variables. Various data and multimedia formats and technologies allow the display of animated web maps: SVG, Adobe Flash, Java, Quicktime, etc., also with varying degrees of interaction. Examples for animated web maps are weather maps, maps displaying dynamic natural or other phenomena (such as water currents, wind patterns, traffic flow, trade flow, communication patterns, etc.).[5]

3.1.5 *Realtime Web Maps*

Realtime maps show the situation of a phenomenon in close to realtime (only a few seconds or minutes delay). Data is collected by sensors and the maps are generated or updated at regular intervals or immediately on demand. Examples are weather maps, traffic maps or vehicle monitoring systems [5].

3.1.6 *Personalized Web Maps*

Personalized web maps allow the map user to apply his own data filtering, selective content and the application of personal styling and map symbolization. The OGC (Open Geospatial Consortium) provides the SLD standard (Styled Layer Description) that may be sent to a WMS server for the application of individual styles. This implies that the content and data structure of the remote WMS server is properly documented [5].

3.1.7 *Open, Reusable Web Maps*

Web maps in this category are usually more complex web mapping systems that offer APIs for reuse in other people's web pages and products. An example for such a system with an API for reuse is Google Maps with the Google Maps API [5].

3.1.8 *Interactive Web Maps*

Interactivity is one of the major advantages of web maps. It helps to compensate for the disadvantages of screen and web maps. Interactivity helps to explore maps, change map parameters, navigate and interact with the map, reveal additional information, link to other resources, and much more. Technically, it is achieved through the combination of events, scripting and DOM manipulations [5].

3.1.9 *Analytic Web Maps*

These web maps offer GIS analysis, either with geodata provided, or with geodata uploaded by the map user. As already mentioned, the borderline between analytic web maps and web GIS is blurry. Often, parts of the analysis are carried out by a serverside GIS and the client displays the result of the analysis. As web clients gain more and more capabilities, this task sharing may gradually shift [5].

3.1.10 *Online Atlases*

Atlas projects often went through a renaissance when they made a transition to a web based project. In the past, atlas projects often suffered from expensive map production, small circulation and limited audience. Updates were expensive to produce and took a long time until they hit the public. Many atlas projects, after moving to the web, can now reach a wider audience, produce cheaper, provide a larger number of maps and map types and integrate with and benefit from other web resources [5].

3.1.9 *Collaborative web maps*

Collaborative maps are still new, immature and complex to implement, but show a lot of potential. Technically, an application allowing simultaneous editing across the web would have to ensure that geometric features being edited by one person are locked, so they can't be edited by other persons at the same time. Also, a minimal quality check would have to be made, before data goes public [5].

3.2 Advantages of web maps

Web maps can easily deliver up to date information. If maps are generated automatically from databases, they can display information in almost realtime. A good example would be a map displaying the traffic situation near realtime by using traffic data collected by sensor networks.

Because web maps distribute both logic and data with each request or loading, product updates can happen every time the web user reloads the application. In traditional cartography, when dealing with printed maps or interactive maps distributed on offline media (CD, DVD, etc.), a map update caused serious efforts, triggering a reprint or remastering as well as a redistribution of the media. With web maps, data and product updates are easier, cheaper, and faster, and can occur more often [13].

If web maps are implemented based on open standards, the underlying operating system and browser do not matter as long as they support the standard. Using open standards and documented APIs, one can integrate (*mash up*) different data sources, if the projection system, map scale and data quality match. The use of centralized data sources removes the burden for individual organizations to maintain copies of the same data sets.

Web maps allow for personalization. By using user profiles, personal filters and personal styling and symbolization, users can configure and design their own maps, if the web mapping systems supports personalization. Accessibility issues can be treated in the same way. If users can store their favorite colors and patterns they can avoid color combinations they can't easily distinguish (e.g. due to color blindness).

Web maps support hyperlinking to other information on the web. Just like any other web page, web maps can act like an index to other information on the web. Any sensitive area in a map, a label text, etc. can provide hyperlinks to additional information [6].

3.3 Disadvantages of web maps and problematic issues

Many web maps are of poor quality, both in symbolization, content and data accuracy. Despite the increasing availability of free and commercial tools to create web mapping and web GIS applications, it is still a complex task to create interactive web maps. Many technologies, modules, services and data sources have to be mastered and integrated [6].

The reliability of the internet and web server infrastructure is not yet good enough. Especially, if a web map relies on external, distributed data sources, the original author often cannot guarantee the availability of the information.

Web maps usually need a relatively high bandwidth. Like with other screen based maps, web maps have the problem of limited screen space. This is in particular a problem for mobile web maps and location based services where maps have to be displayed in very small screens with resolutions as low as 100×100 pixels.

Compared to the development of standalone applications with integrated development tools, the development and debugging environments of a conglomerate of different web technologies is still awkward and uncomfortable.

Chapter 4

System Analysis

The objective of the project is to develop a web site that provides map based driving route advisor for Addis Ababa City. In order to achieve the envisaged objective, Environmental Systems Research Institute (ESRI)[i] GIS serve model and its associated Web Application Development Framework (Web ADF) are employed for developing the solution.

4.1 Functional requirements

4.1.1 *Driving/Navigation Route Map*

The system when supplied two point addresses, namely the starting and the destination point information, displays a map of Addis Ababa City that shows the optimum route between the two points. For ease of operation the two points must be named after known places (business) along the street. This is because since street addressing scheme of Addis Ababa is not standardized it doesn't lend itself for geocoding[ii] process.

4.1.2 *Driving/Navigation Route Map for Reverse Direction*

The system, up on request, delivers the appropriate reverse route map for every route map it generates (i.e. for each solutions delivered for the above functionality). When producing the reverse route map the system should consider sets of navigation restriction applicable (e.g. one way road) in analyzing the reverse route advice.

4.1.3 *Driving/Navigation Route Guide*

In all of the above route advice map presentation function, the system should also display navigation advice in tabular form that associates with the resulting route advice. The map and the tabular information must much in terms of how long to travel along a given edge (street) of the resulting route and has to tell which direction of turn to take when a given edge (street) end is reached.

---

[i] *One of the leading GIS technology companies*
[ii] *Geocoding is a process of identifying a point in a map (e.g. on street network map). It helps to know the starting and destination points in the street network. Given this two points it iss up to the network analysis algorithm to find the path between this two points with by considering the minimum possible distance between them.*

4.1.4 *Driving/Navigation Route Print Out*

The system must have a function that serves to print the map displayed and if need be the navigation advice table together for future reference.

4.1.5 *Upload Business Location Information*

Enable users to enrich the street network database by allowing them to upload business location and share them with others.

4.1.6 *Upload and Use Traffic Information*

Enable system administrators to enrich the street network database by allowing them to upload real time traffic information supplied in XML format..

4.1.7 *Integrate with Google Map*

Integrate the site with known mapping and business service search provider-Google in order to boost the user experience.

4.2 Nonfunctional requirements

4.2.1 *User Interfaces*

The user interface for Online Route Finder should satisfy the following properties.

- ☞ It should be consistent across all the services
- ☞ It should be menu driven
- ☞ It should be easy to use and intuitive

4.3.1 *Security*

The system should harness the GIS Server security system supplied with the GIS server itself. The GIS server uses three accounts: the ArcGIS Web Services Account, the Server Manager Account and the Server Content Account.

The ArcGIS Web Services Account is used to process Web service requests on the GIS server. This account is used internally by the Web server to communicate with the GIS Server on behalf of a user when a user makes an Internet connection and requests a route advice.

The Server Object Manager (SOM) account is the account that runs the GIS Server. The manager manages the GIS server and has privileges to start/stop the GIS server, create/delete/edit GIS services and grant/deny/revoke users access to server resources.

Server Object Container (SOC) account is the one given to content authors/editors who create content that will ultimately be posted on GIS server Therefore, the SOC account must have read access to any GIS resources (maps, locators, data) that any preconfigured and application-specific services require to do their work. In addition, the SOC Account must have write access to the server directories.

4.3 Analysis Model

To produce a model of the system which is correct, complete and consistent it is imperative to construct the analysis model which focuses on structuring and formalizing the requirements of the system. Analysis model contains three models: functional, object and dynamic models. The functional model can be described by use case diagrams. Class diagrams can describe the object model. Dynamic model can also be described in terms of Sequence, state chart and activity diagrams. For the purpose of this project; the analysis model is described in terms of the functional model and dynamic models using use case, sequence and activity diagrams

4.3.1 *Use case Diagram*

Use case (Figure 1) identified for the project includes *"Driving Direction", "Reverse Driving Direction", "Print Routing Advice", "Upload Business Information", "Display Route in Google Map", "Author/Edit Network Analysis Map" and "Create/Update Geocoding Information"*. The first five use cases are initiated by the Internet users while the rest two are initiated by the system administrator.

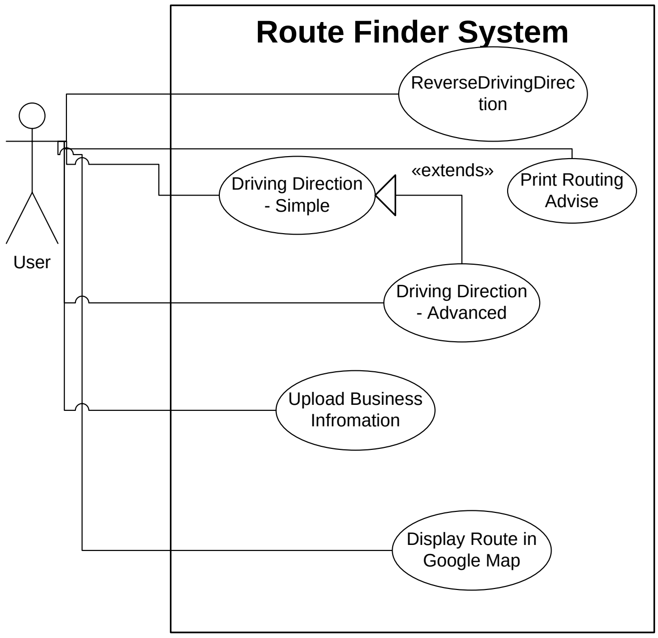


Figure 1: Use Case Model

Below are the tables describing the sequence of actions executed by each use cases.

Use Case Name	**DRIVING DIRECTION-SIMPLE**
Entry Condition	None
Flow of Events	1. Users type in the URL of the application into their browser 2. Route Finder System will display home page 3. Users select starting location are form a drop down list which providing possible starting locations 4. The system generates possible starting places that are found in the selected location and displays to user using dropdown list 5. From the drop down list of possible start locations users select one starting location 6. Users select destination location are form a drop down list which providing possible destination locations 7. The system generates possible destination places that are found in the selected location and displays to user using

	dropdown list 8. From the drop down list of possible destination locations users select one destination location 9. The user press get direction button 10. Route Finder System generates routing map based on the entry
Exit Condition	Routing map and navigation advice delivered to users browser

Table 1 Table driving direction – simple use case description

Use Case Name	**DRIVING DIRECTION-ADVANCED**
Entry Condition	None
Flow of Events	1. Users type in the URL of the application into their browser 2. Route Finder System will display home page 3. Users select starting location are form a drop down list which providing possible starting locations 4. The system generates possible starting places that are found in the selected location and displays to user using dropdown list 5. From the drop down list of possible start locations users select one starting location 6. Users select destination location are form a drop down list which providing possible destination locations 7. The system generates possible destination places that are found in the selected location and displays to user using dropdown list 8. From the drop down list of possible destination locations users select one destination location 9. The users select their traffic tolerance preference (there level high, medium and low) 10. The user press get direction button 11. Route Finder System generates routing map based on the entry
Exit Condition	Routing map and navigation advice delivered to users browser

Table 2: Table driving direction – advanced use case description

Use Case Name	**REVERSE DRIVING DIRECTION**
Entry Condition	Routing map and navigation advice is delivered to users
Flow of Events	1. Users click on 'Reverse Direction' button on the result page 2. Route Finder System generates reverse routing map and navigation advice based on the previously delivered routing request entries
Exit Condition	Reverse routing map and navigation advice delivered to users browser

Table 3: Reverse Driving Direction use case description

Use Case Name	**PRINT ROUTING ADVISE**
Entry Condition	Routing map and navigation advice is delivered to users
Flow of Events	1. User click on 'Print' Button on result page 2. Route Finder System generates printable version of the current result in new window
Exit Condition	Map with navigation advice ready for printing

Table 4: Print routing advice use case description

Use Case Name	**UPLOAD BUSINESS INFORMATION**
Entry Condition	None
Flow of Events	1. Users click on "New Business" button 2. Route Finder System opens New Business information form in the browser 3. Users enter appropriate entry into the form and click submit button 4. Route Finder System stores the data into the geodatabases
Exit Condition	New Business information data stored in the geodatabases

Table 5: Upload business information use case description

Use Case Name	**DISPLAY ROUTE IN GOOGLE MAP**
Entry Condition	Business Search Result
Flow of Events	1. Users click on “Search Google” button 2. Will probe Google map service and displays matching result on respective mapping controls implemented with their Map API
Exit Condition	Route is displayed in side Google map

Table 6: Display route in Google Map use case description

4.3.1 *Sequence Diagram*

Sequence diagrams show the interaction between participating objects in a given use case. They are helpful to identify the missing objects that are not identified in the analysis object model. To see the interaction between objects, the following describe the sequence diagram of each identified use cases

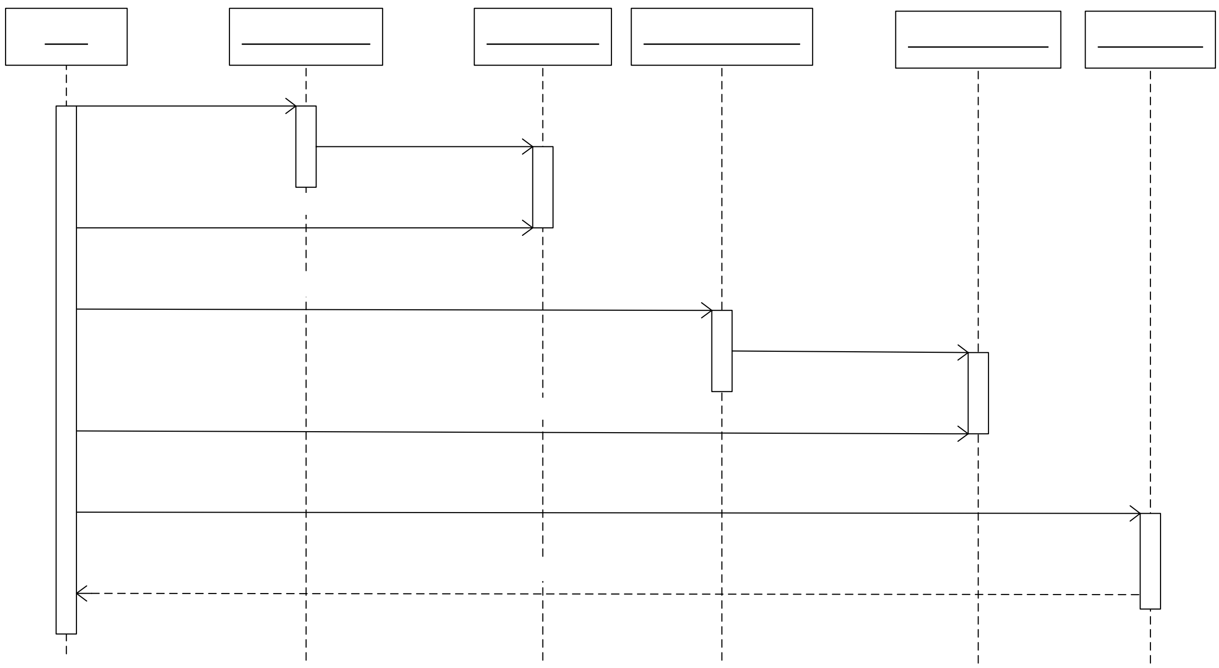


Figure 2: Sequence diagram for simple driving direction advice

:User :StartingLocality :St

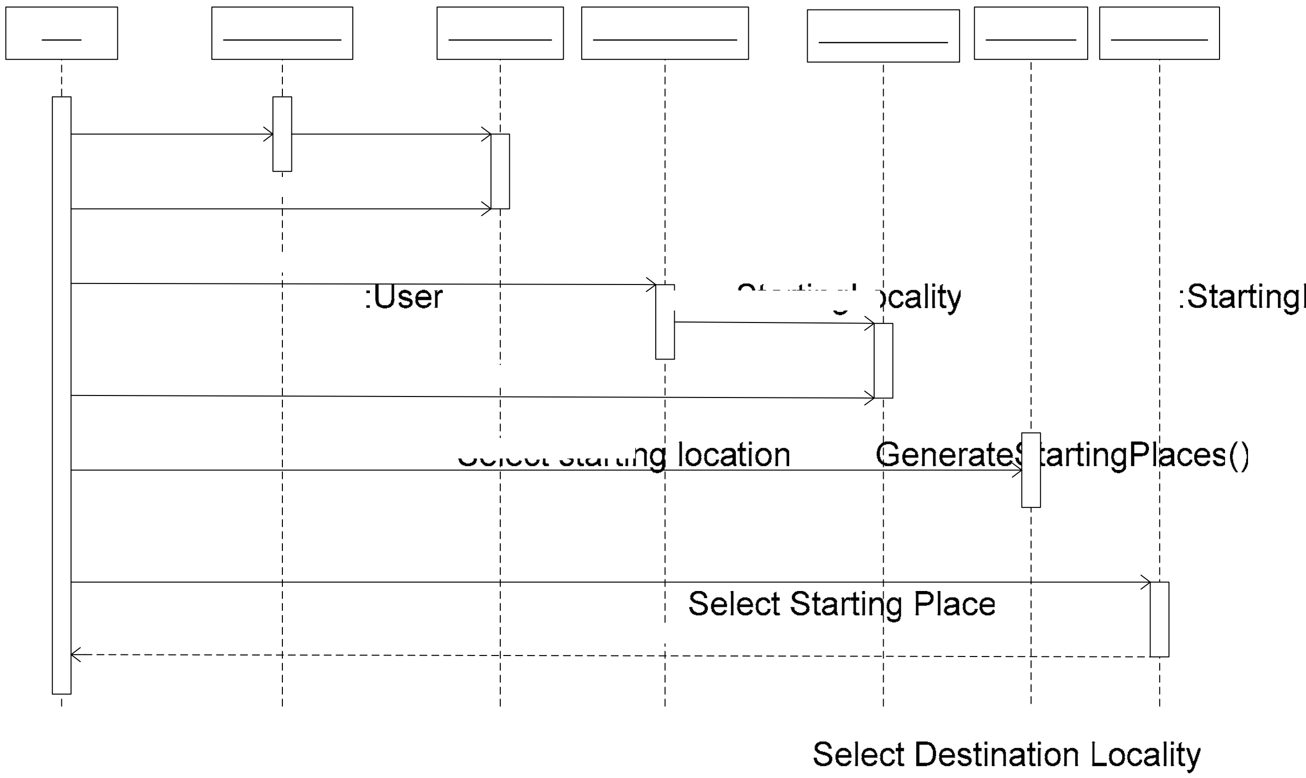


Figure 3: Sequence diagram for advanced driving direction advice

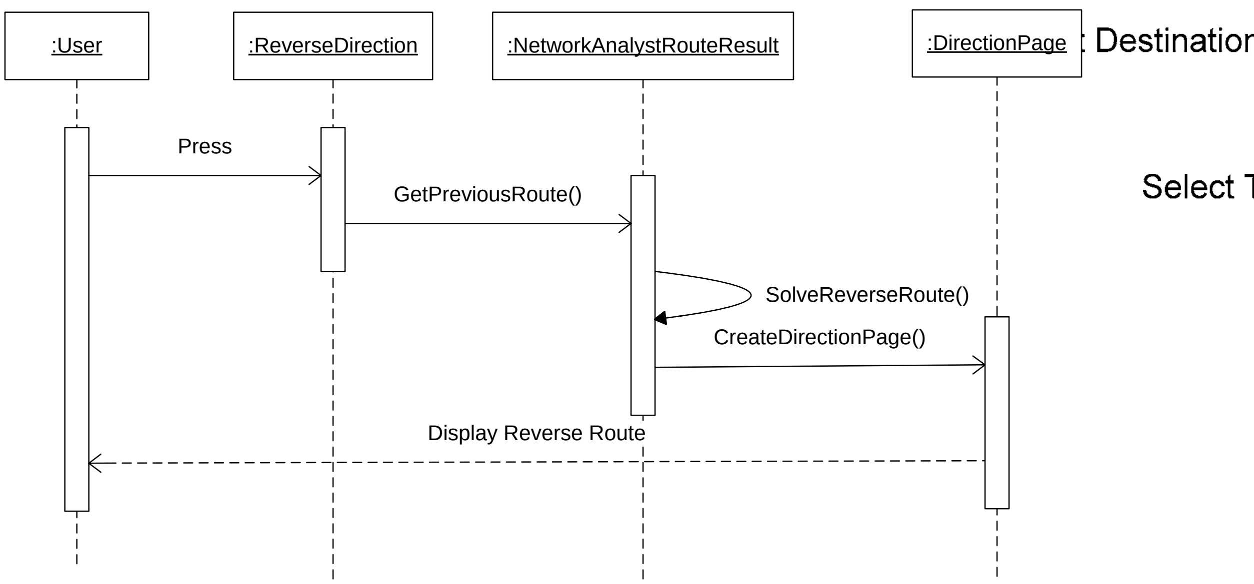


Figure 4: Sequence diagram for reverse direction advice

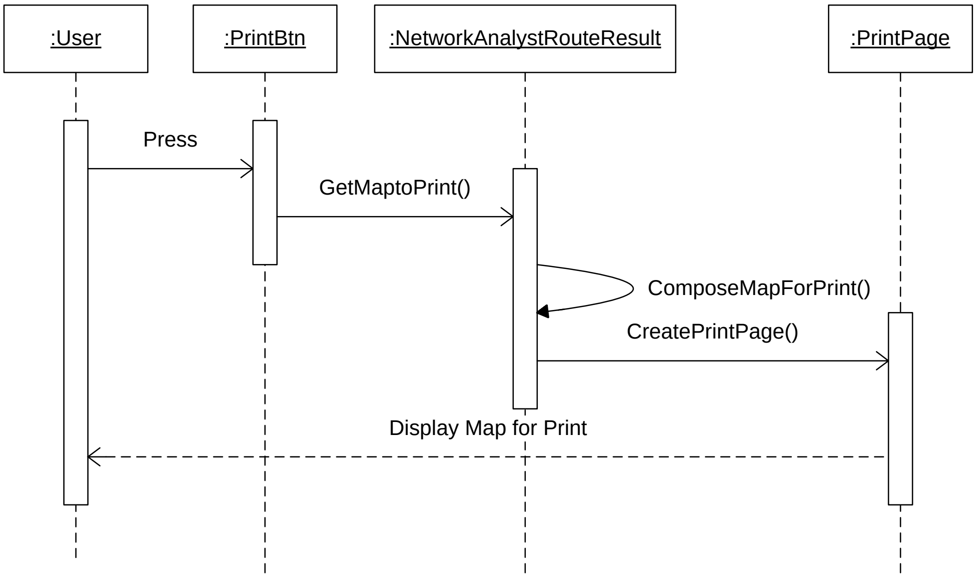


Figure 5: Sequence diagram for route advice printing

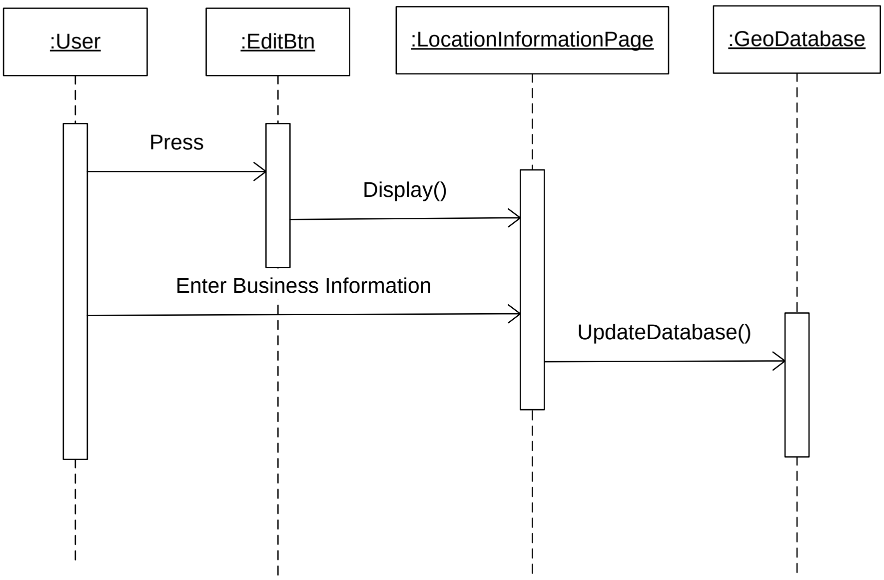


Figure 6: Sequence diagram for business information upload

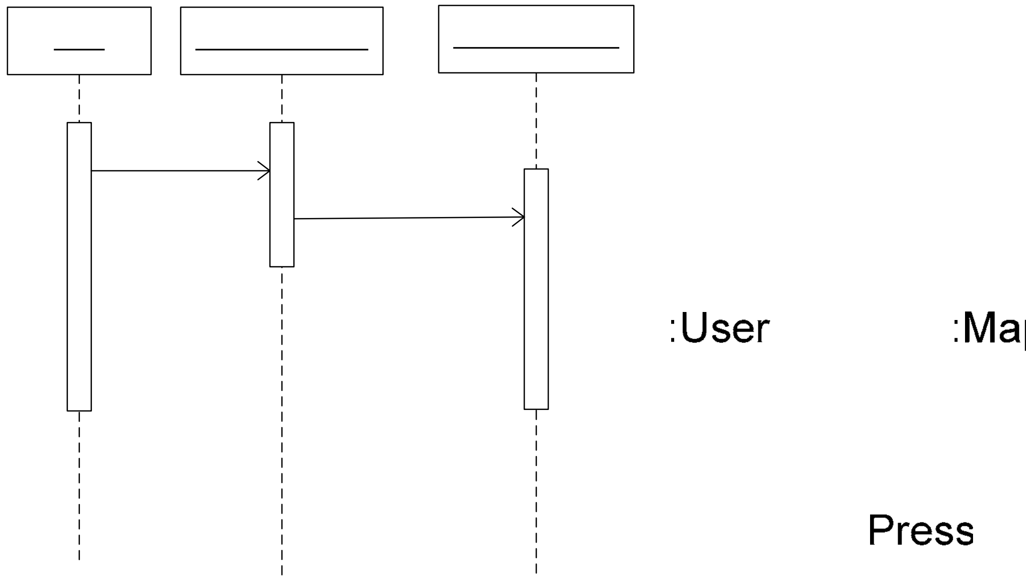


Figure 7: Sequence diagram for uploading route information to Google Map

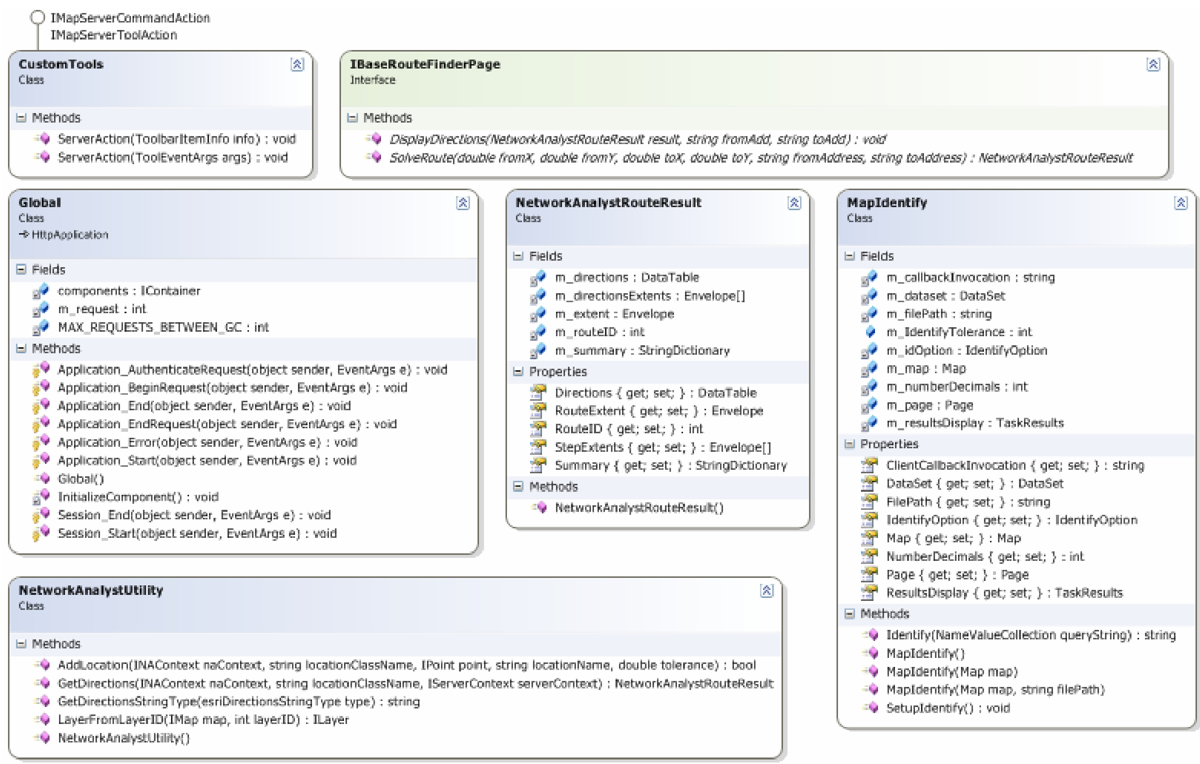


Figure 8: Class Diagram

Chapter 5

System Design

In the previous chapter the functional and non-functional requirements of the system were identified and the analysis model was produce. Based on these considerations, the design of the system is presented in this chapter. Before embarking the detail discussion, let's investigate the design goals. Following that the architecture of the system will be described.

5.1 Design Goals

In implementing web mapping response time for each request must be optimal. As mentioned in chapter 3, bandwidth, capacities of the browser in supporting available protocols and client machines local resource has greater impact in web mapping application design and implementation. Hence, the application architecture and the platform it runs on is determined inline with these two points. Besides, system decomposition and logical and physical layout is done in context of the above

The constraints associated with the project that has to be addressed in designing process includes resource limitation on the server side for intensive computation of routing advice, growing geo-database and map image storage and retrieval issues; generally speaking system throughput issues, are in the picture. On the client side, clients need have to have sufficient enough resources as to screen size, compatible browser and sufficient enough connection bandwidth. So the design is in light with optimization of the above resource.

5.2 Architecture of the System

To implement the project, a specialized server system that meets the envisaged goal to realize the web mapping service is needed. One option that fits the bill is picking one of the readymade GIS servers that support the project's requirement keeping in mind that the sole purpose of the server would be to host mapping service and serve it out to client applications upon requests. It will be an add on if the mentioned server provides user friendly set of tools and consol that makes it possible to administer/manage services that are installed/staged on it.

Such kind of features enables to smoothly execute administrative choruses associated with the mapping servicesi being served.

After assessing the available potential GIS servers and the associated capability of each; the project picked up ESRI's ArcGIS 9.2 server for it fits the bill perfectly. Hence the project architecture is implemented in conformity with what ArcGIS server supports.

An ArcGIS Server system's architecture, which is also the architecture of the application developed in this project, is made up of some of the following components (Figure 9):

i. **GIS server—** The GIS server hosts GIS resources, such as maps and address locators and exposes them as services to client applications. In our project the project map created using ArcMap is published on the server.

   The GIS server itself is composed of two distinct parts: the **server object manager** (SOM) and **server object containers** (SOC). As the name implies, the SOM manages the services running on the server. When a client application requests the use of a particular service, it's the SOM that actually gives one out for the client to use.

   The SOM connects to one or more SOCs. The SOC machines—also referred to as container machines—contain, or host, the services that the SOM manages. Depending upon the configuration, the SOM and SOC can on the same or different machines.

ii. **Web server—** The Web server hosts Web applications and Web services that use the resources running on the GIS server. The application of this project is developed using ASP.NET and posted on this server to be served for browser

iii. **Clients—** Client applications are Web applications that connect to ArcGIS Server Internet services (or ArcGIS Server) using Hypertext Transfer Protocol (HTTP) over a LAN or Internet[ii]. The client was developed using ASP.NET 2.0.

---

[i] *Here services are GIS resources like map and geo-processing functionalities which have been made available on the server are collectively referred to as services.*

[ii] *For our case we make use of Web application that run on a LAN setting*

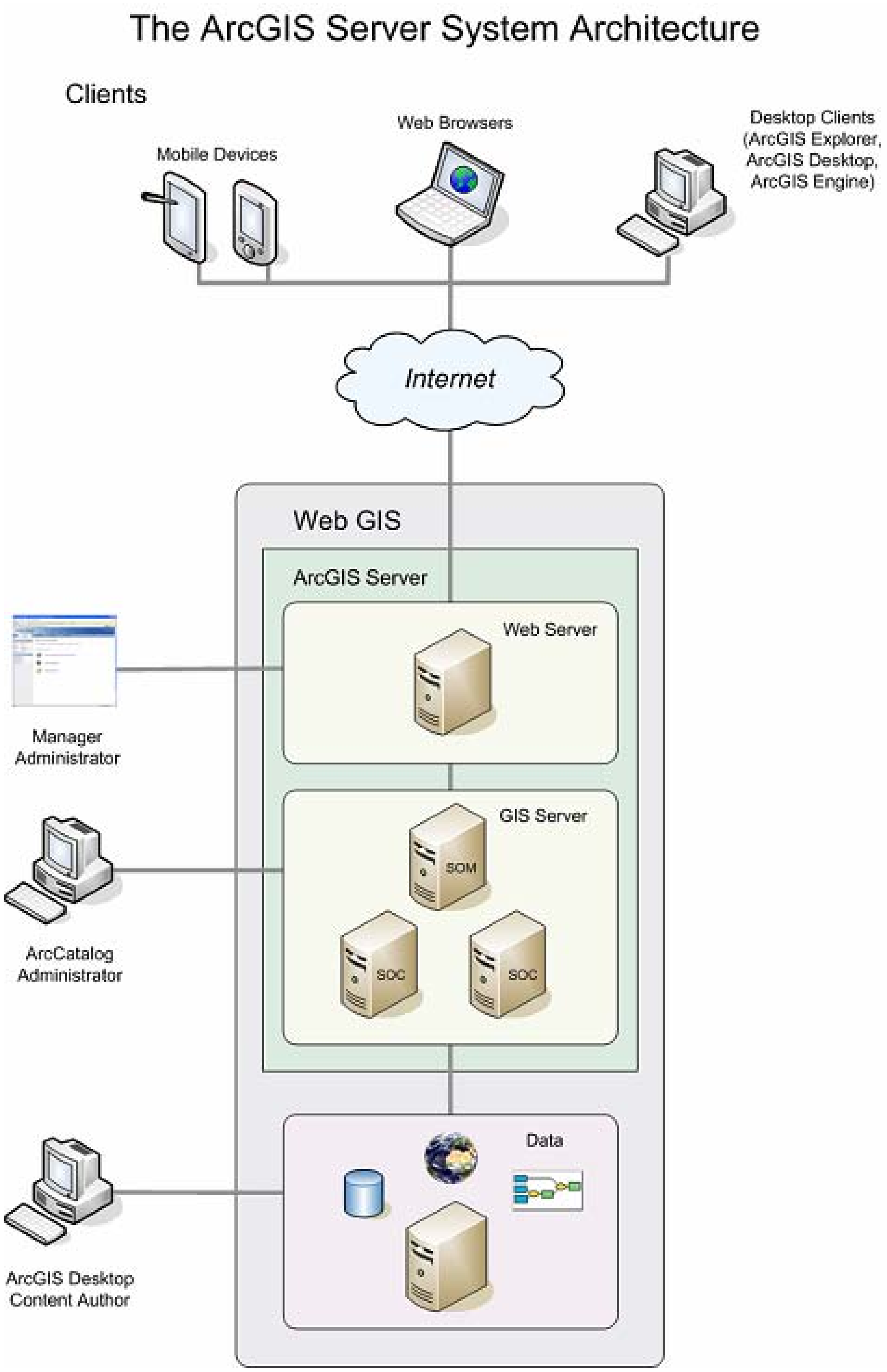


Figure 9: System Overall Architecture

iv. **Data server—** The data server contains the actual GIS resources that have been published as services on the GIS server. These resources can be map documents, address locators, geodatabases, and toolboxes.

v. **ArcGIS Server Manager and ArcCatalog administrators–** ArcGIS Server administrators can use either ArcGIS Server Manager or ArcCatalog to publish their GIS resources as services.

ArcGIS Server Manager[i] (Figure 10) supports publishing services, administering the GIS server, creating Web applications, and publishing ArcGIS Explorer maps on the server.

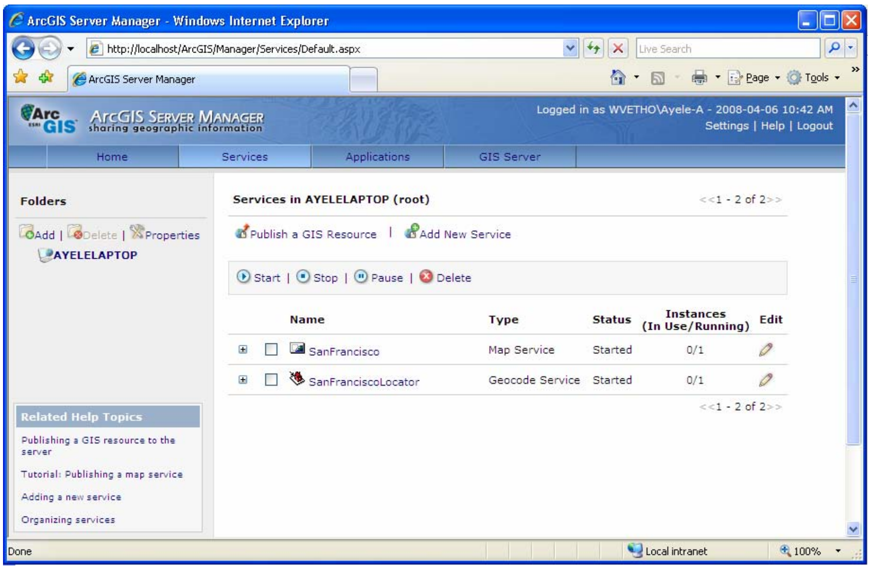


Figure 10: ArcGIS Server Manager

ArcCatalog (Figure 11) includes a GIS Servers node which can be used to add connections to GIS servers for either general server usage or administration of a server's properties and services.

[i] *ArcGIS Server Manager console itself is a web application*

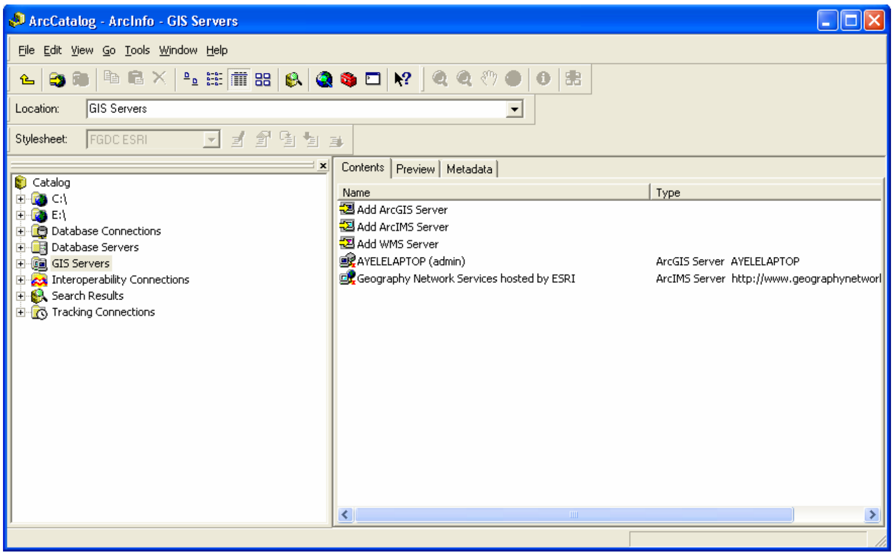


Figure 11: ArcCatalog

vi. **ArcGIS Desktop content authoring Tools**–To author the GIS resources such as maps, geoprocessing tools, and globes that will be published to the server, ArcGIS Desktop applications such as ArcCatalog (Figure 11), ArcMap/ArcInfo (Figure 12) and ArcGlobe can be used.

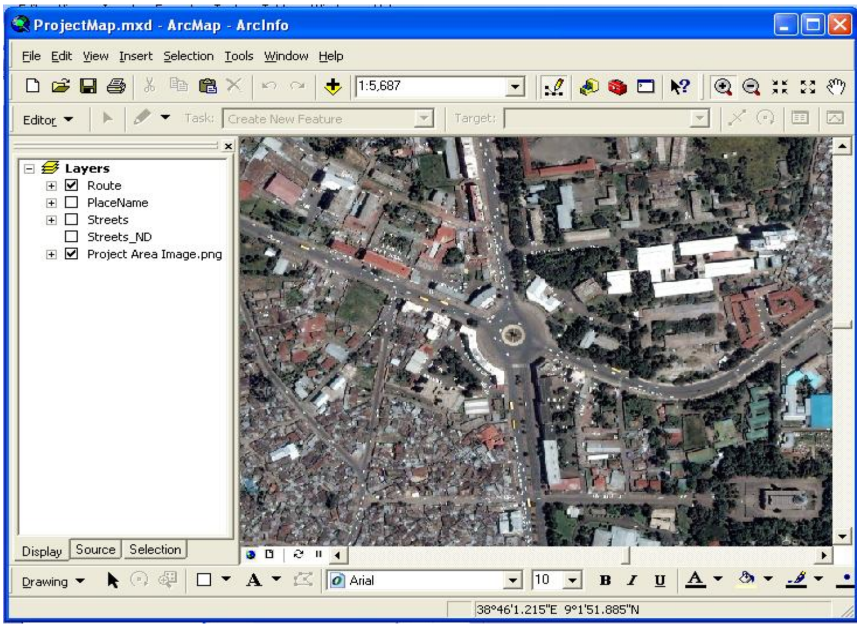


Figure 12: ArcInfo

5.3 Algorithm

As mentioned in section 5.2 the server technology employed is ArcGIS. This server makes use of the Network Analyst library for working with network datasets. Figure 13 shows an overview of some of the objects in the Network Analyst library.

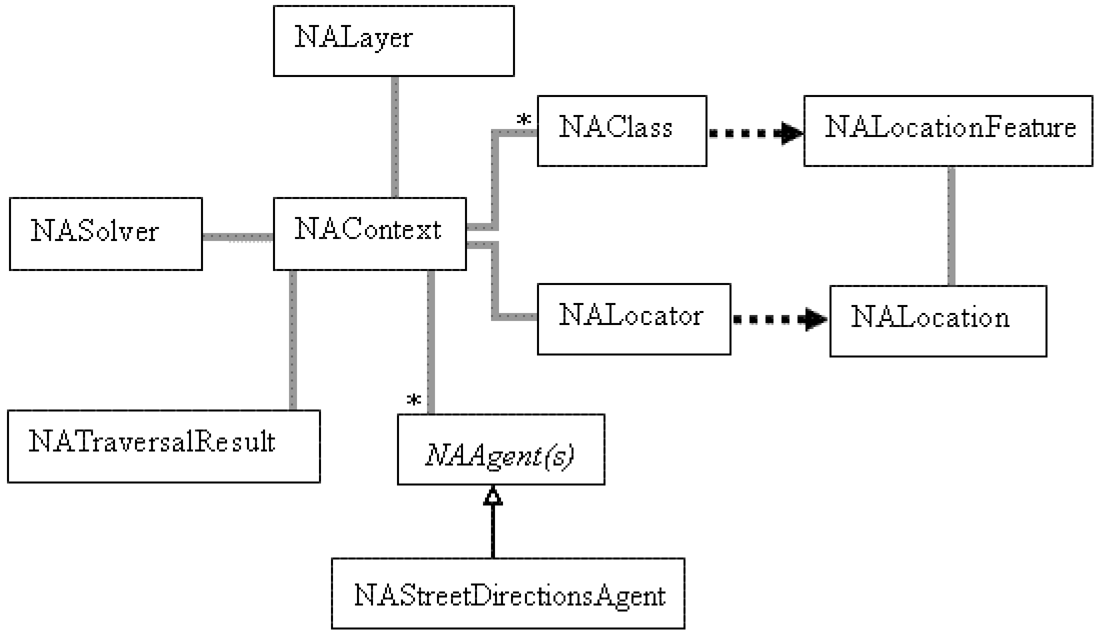


Figure 13: Network Analyst Diagram

As it can be seen in figure 13, **NAContext** is at the center of the object model. NAContext is the primary object used during network analysis. It is used to save the various parameters that any network solver needs when performing network analysis. Examples of parameters that are maintained by the NAContext are a route and the location of stops or barriers.

Any map document that is used for network analysis must contain a network analyst layer in this project case the network layer would be the street network shape. The network analyst layer gives access to the rest of the objects that will be used in the application. The NAContext is accessed by getting the Context property from the NALayer. The NALayer is a composite layer that holds a reference to the NAContext and exposes the NAClasses within the NAContext as feature layers.

The NAContext provides access to the NetworkDataset, NASolver, NAClasses, NATraversalResult, NALocator, and any extra objects that implement INAAgent. The NAClass is an in-memory feature class holding features used as input and generated as output during network analysis. The NATraversalResult object holds the results of the network analysis. The NALocator is used to find the NALocations which specify the side and position along a feature in a Network Dataset.

It is the usage of NAContext that is key to all Network Analyst applications. The next section describes the purpose of the NAContext and how it can be used to maintain the state of a Network Analyst application.

It is expected that many users will be interacting with the web application and hence maintain each user's changes is mandatory. The MapDescription is specifically designed to support the programming patterns needed to employ to save state and apply it to the map. The NAContext is designed to support the saving of state for Network Analyst applications.

Just like any other application, a Network Analyst application generates results that are specific to a particular user. For example, a user might enter two addresses and find directions between the two points. Another user will create directions from another set of locations. Thus, the web application needs to maintain some information about each client. By hosting the map that contains the network layer as a pooled object, clients will share the object. Thus, the server must maintain the state of what each client is doing. It is the NAContext that is used to maintain the state of the Network Analyst application. Therefore; each user of the application has their own copy of NAContext.

When creating a Network Analyst application that accesses ArcGIS Server, the NAContext is serialized in order to store all Network Analyst state from one request to the other. For example, once the application has displayed a route on the map, there might be a need to continue to display that route even after the map has been panned or zoomed. For each request to the application, a process occurs to retrieve, store, copy or apply, serialize, and finally restore the NAContext which has this information. Typically, during each use of a map server object, i.e., each request from the Web browser, the following steps take place for each Network Analyst layer.

1. At the beginning of the request, retrieve the original NAContext from the network analyst layer.
2. Save the original NAContext.

   If this is the first request,

   a. Make a copy of the NAContext by serializing and deserializing it.
   b. Store this copy for the duration of the request.

   If this is not the first request, apply the copied NAContext saved in the previous request in order to restore the Network Analyst state.

3. Perform the necessary Network Analyst work such as adding a network location and solving for a route.
4. At the end of the request, serialize the current NAContext and store it for the next request.
5. Restore the original NAContext retrieved at the beginning of the request to return the server object to its initial state.

When the NAContext is serialized, it serializes all of its member objects. For instance, all of the stops, barriers, and facilities in the NAClasses of that Network Analyst layer will get serialized. This is an important consideration when working with a Network Analyst layer that has a potentially large number of facilities as the serialized NAContext can get very large. Once the NAContext gets large, there may be impacting the performance of the web application.

5.4 User Interface Design

5.4.1 *The Home Page*

The main web page (Figure 14) is the one which is displayed when the users type in the URL of the site into their browsers address bar. It comprises three sections

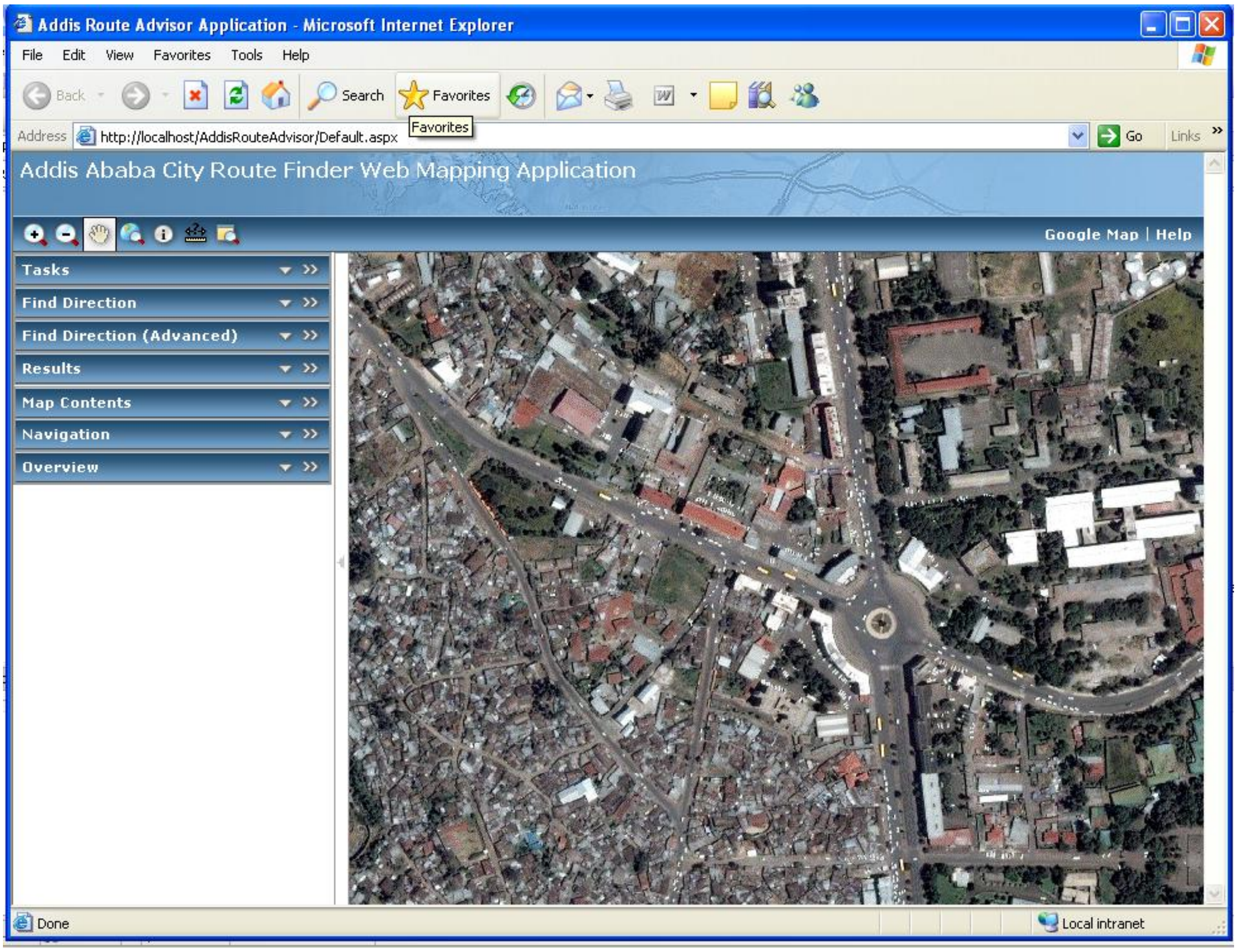


Figure 14: Main Web Page

Toolbar (Figure 15): Underneath the title of the site there exist a series of button mainly used zoom in, zoom out, pan, see full extent, information, distance measurement and magnify according to their order of appearance form left to right.

Figure 15: Toolbar

Menu System: (Figure 16) the menu system provides an entry point for the functionally of the map.

Figure 16: Menu System

Task: Provides access to business searching and uploading business information.

Find Direction (Simple): Provides a user interface to enter starting point and destination point used to analyze direction without considering traffic preference. Here, the default traffic tolerance assumed for detrimining the route is to be high. Hence, the suggetsted root will pass through heavy traffic streets. Figure 17 shows the route suggested for user who wants to go from Joly Bar (4 kilo) to Anbesa Gebi (5 kilo)

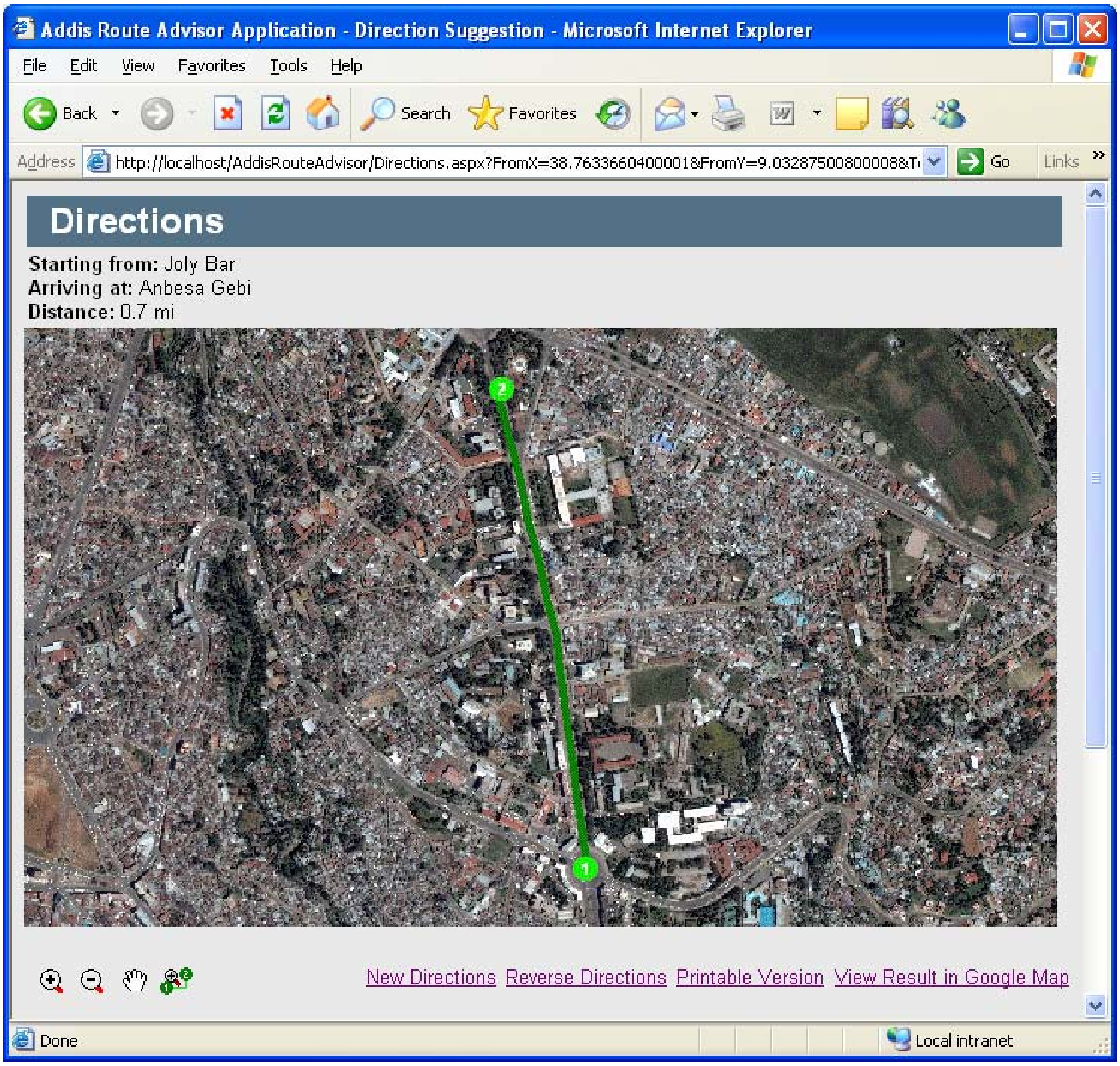


Figure 17: Driving Direction Advice Between Joly Bar and Anbesa Without Considering Traffic Level

Find Direction (Advanced): Provides a user interface to enter starting point, destination point and traffic tolerance. The traffic tolerance is the magnitude of tolerance that the user tolerates regarding traffic density. The tolerance is expressed as high, medium or low. High tolerance means the user tolerates the traffic no matter what it is dense and low means the user does not tolerate traffic as such and seeks low traffic density routes available. Figure 18 shows the result for the same query presented to the simple direction suggestion between Joly Bar and Anbessa Gebi (Figure 17) but by setting traffic preference to medium.

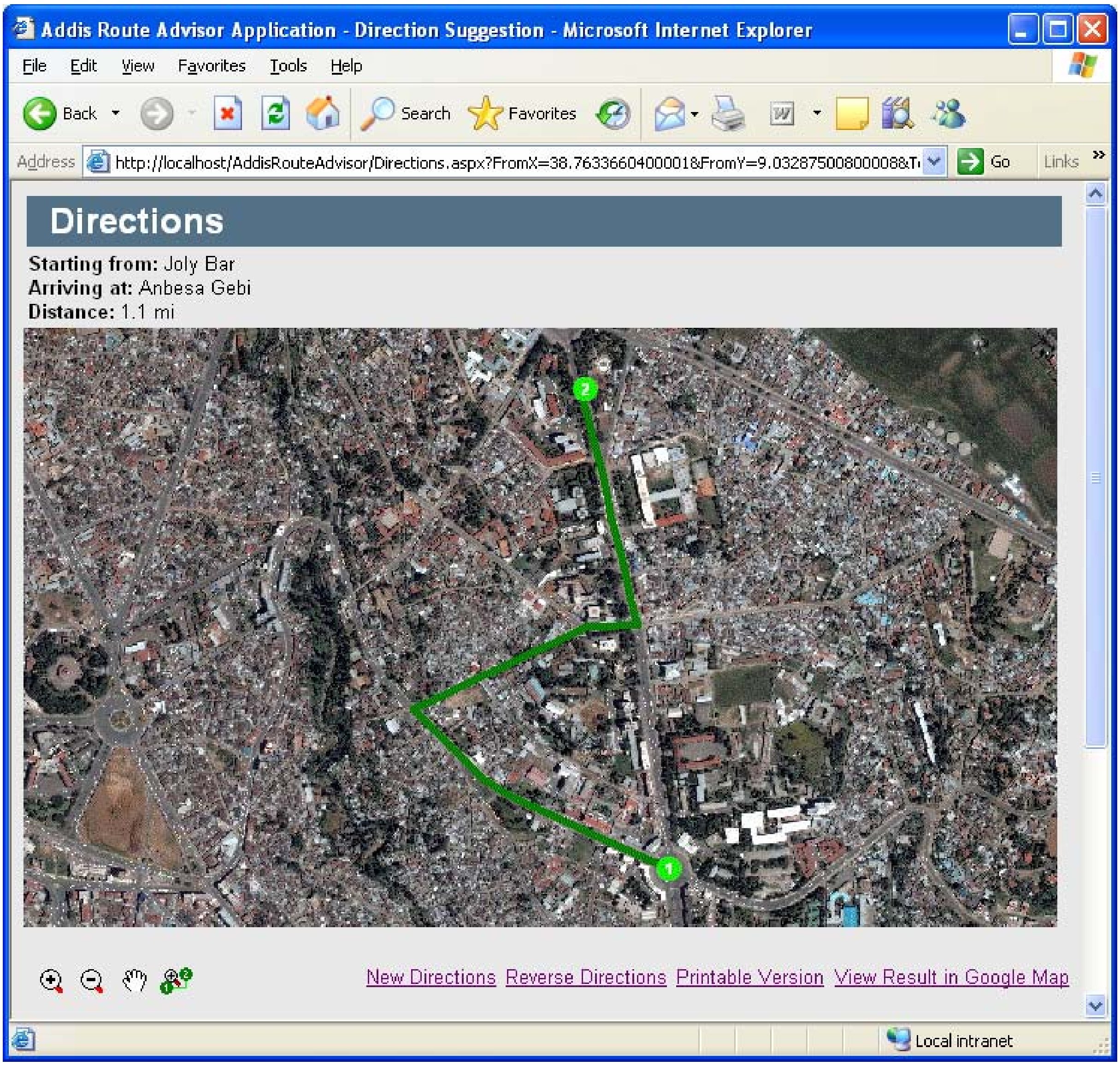


Figure 18: Driving Direction Advice between Joly Bar and Anbesa by Avoiding High Traffic Street

Result: Provides place to display search result and identify information.

Map Content: Provides table of content to the map layer displayed in the map window.

Navigation: Provides gives support in navigating around the map.

Overview: Provides visual clue as to which area currently is displayed in map window as compared to the entire map.

Map window: (Figure 19) It is used to display the map

Figure 19: Map Window

5.4.2 *The Direction Page*

The direction page is used to display the result of route analysis. The page has two sections. The upper section, as shown in figure 16 and 17, displays the suggested route superimposed on satellite map. The lower section contains textual navigation advice and expected length of each edge arranged in tabular form as shown in figure 20.

Driving Directions

Step	Directions	Length	Cumulative Length
1	Start at Joly Bar. Go North on [Arat Kilo Square - Mariam Church] toward [Mariam Church -Amist Kilo]/[Mariam Church]/[Mariam Church - Good Sheepered]	0.3 mi	0 mi
2	Continue on [Mariam Church -Amist Kilo]	0.2 mi	0.3 mi
3	Continue on [Amist Kilo - Yekatit 12 Hospital]. Finish at Anbesa Gebi, on the right	0.1 mi	0.6 mi

Figure 20: Driving Direction Page

At the top of the page, it gives brief information about the starting and arriving destination places and the total distance elapsed between these two points. The route is displayed in the map and the map can be zoomed in or out, panned and seen to full route extent using the tools just below the map. The hyperlink button gives for further functionalities as described in the use case model. The detail of is as follows

New Direction: This will enable the user to go to home page reset the session so that new route advice request can be made

Reverse Direction: The system will make the possible reverse direction analysis by considering and restriction on the way

Printable Version: Delivers a printable version of the map

View Result in Google Map: This will mash up the route analysis result into Google Map site.

Chapter 6

Development

6.1 Programming Tool

To develop client interface Web Application Developer Framework (ADF) for the Microsoft .NET Framework which is provided by ESRI is used. The Web ADF itself is built on top of the Microsoft .NET Framework by extending the .NET Framework class library with new classes that support a set of custom web controls which in turn provides local and remote access to supported data sources. Figure 21 shows how the Web ADF fits into the overall development environment.

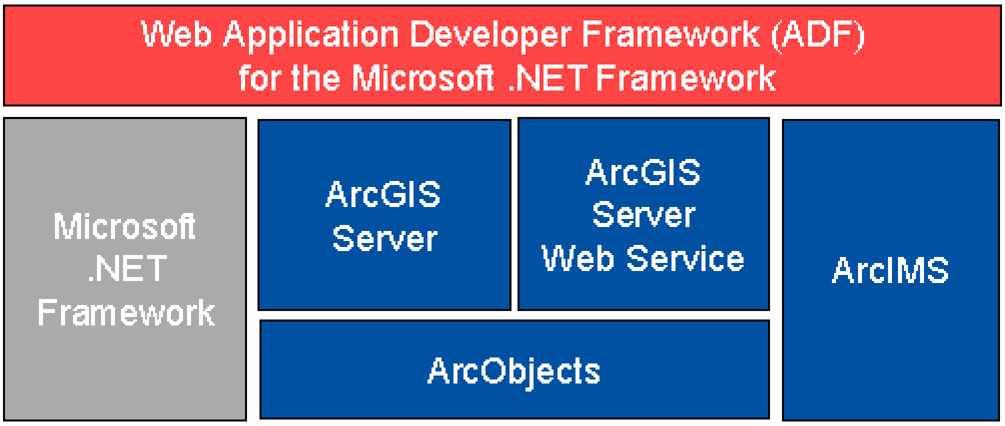


Figure 21: Web Application Developer Framework Architecture

The ADF basically serve as a bridge between the .NET Framework; particularly ASP.NET technology and ArcGIS server class library. ASP.NET technology is a set of class libraries that support the development of web sites and web services on Microsoft's .NET Platform. On the other hand ArcGIS server hosts and serves map services, geocoding services, KML services, network analysis services, which are authored using ESRI's GIS content authoring tools like ArcMap/ArcInfo, ArcMap/ArcView, ArcGlobe, etc… The core functionality of the ArcServer is implemented using ArcObject class library. This class library delivers core GIS and cartographic functionality.

The ADF is there to serve as an interface between the underling ArcGIS server and the web controls so that the complex geo-processing services of ArcObject class can be easily consumed by web controls. In order to facilitate the development of web sites that consumes web map services, ESRI has created new set of web controls that can be plugged into web page and easily delivered to browsers. These specialized controls are bundled with the ADF and will be installed into any existing Visual Studio suit when the ADF is installed.

It has to be noted that any web mapping application, be it web service or web site that use this platform will ultimately use the ArcObject class library. The web ADF is there to abstract site developers from the details of ArcObject implementation. This enables the developers to focus on client interface development. Any GIS related analysis chorus will ultimately be handled by ArcObject that runs on the ArcGIS server.

In realizing the project three sets of software packages were used. To develop client side user interface Visual Studio 2005 that is equipped with ESRI's Web Application Developer Framework for Microsoft .NET Framework is used. To author the base map of the project area, which includes the street network, ArcMap/ArcInfo is used. To deploy the map on a server as a map service ArcGIS server is used.

6.2 Development

6.2.1 *Google Map Satellite Image Extraction*

To serve as a background for the map, satellite image of the study area was downloaded from Google Map Server. To download the image an HTML document embodied with the following script was submitted to the Google map server.

```
<iframe width="200" height="200" frameborder="0"
scrolling="no" marginheight="0" marginwidth="0"

src="http://maps.google.com/maps?f=q&amp;hl=en&amp;geocode=
&amp;q=&amp;ie=UTF8&amp;ll=10.1000,39.25000&amp;spn=0.00229
9,0.004957&amp;t=k&amp;z=14&amp;output=embed&amp;s=AARTsJqP
ZyAvqRqJyEH2bR3O5XUo1qwd_Q">

</iframe>

<br />

<small>
```

```
<a
href="http://maps.google.com/maps?f=q&amp;hl=en&amp;geocode
=&amp;q=&amp;ie=UTF8&amp;ll=8.9500000,38.75000&amp;spn=0.00
2299,0.004957&amp;t=k&amp;z=18&amp;source=embed"

style="color:#0000FF;text-align:left">View Larger Map</a>

</small>
```

As it can be seen in the above script the **ll** attribute of the anchor (<a>) element is used to specify the latitude and longitude of the center of the image to be retrieved. The value for the latitude and longitude is expressed in decimal degrees.

The image obtained from the Google Map server was in a form of rectangular JEPG file format. Since such kind of simple images cannot be used to as a background for any map; converting the file into consumable geodata through the process of georefenceing must be done. To georeference (change the image into geographic dataset or geodata) four ground control values were collected from field using handheld GPS. These four points were identified by inspecting the four outer most corners of the image and identifying a stable ground reference like building to read its ground latitude and longitude value. After this ground checkpoints was collected, they were plotted on the image which was imported into ArcMap/ArcInfo (Figure 22). Using the check points and ArcMap's georeferencing tool, the image was converted form simple satellite photograph into analyzable raster dataset.

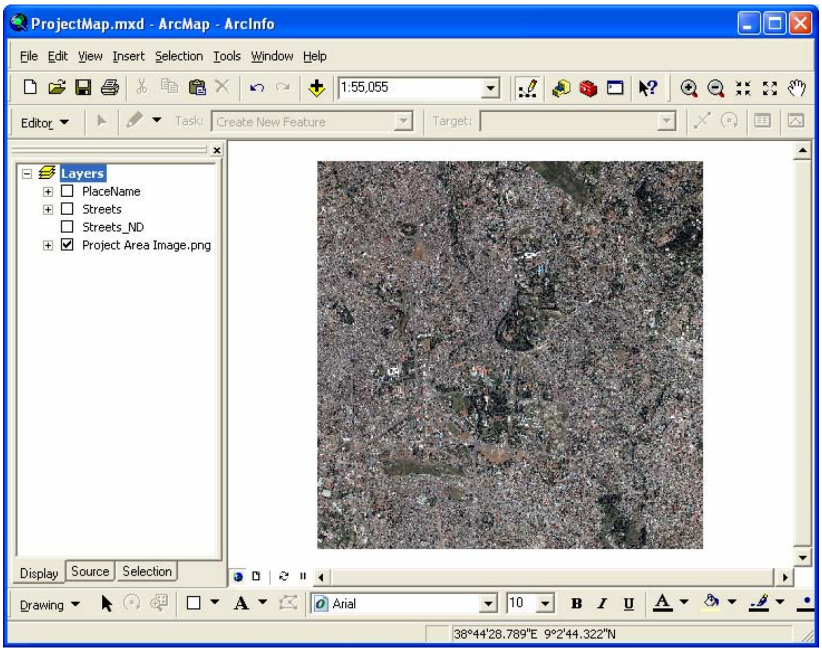


Figure 22: 1:55,055 Scaled Satellite Image of the Project

The map projection chosen during the georeferening process was WGS1984. The projection was picked because many of the popular web mapping service like Google Earth, Google map, Yahoo! Map, Windows Live support that projection. The choice was made intentionally to lessen the data re-projection overhead at driving advice map mash up operation with the mentioned map services.

6.2.2 *Street Digitization*

Streets used in the study were drawn using digitization[i] technique (Figure 23). The above georfereced satellite raster dataset was used as a source of information to determine the path of the streets and their juncture. Since the image resolution was high it lend itself for such work because it was possible to zoom into the image to such an extent that ever street found in the image can be easily traced without loosing any visibility. Actually the accuracy of tracing depends on the accuracy achieved at georeferencing of the base. Besides, the higher the resolution the best is the accuracy of the street network traced using it as a background image.

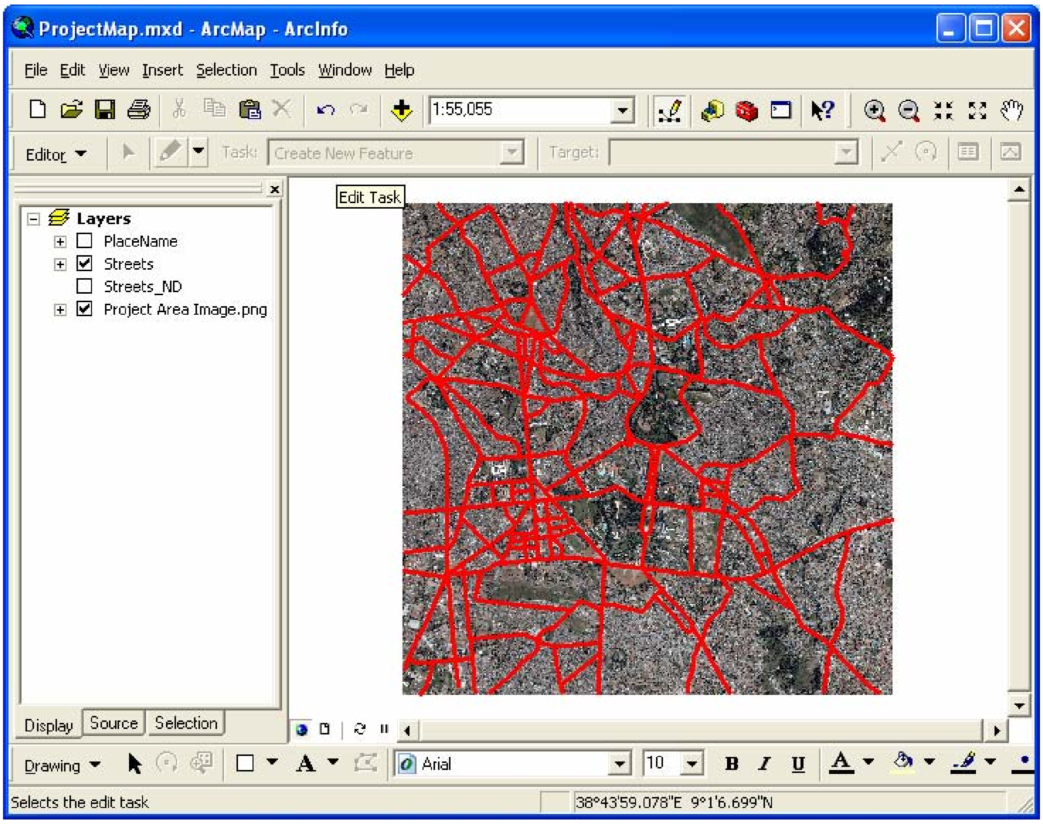
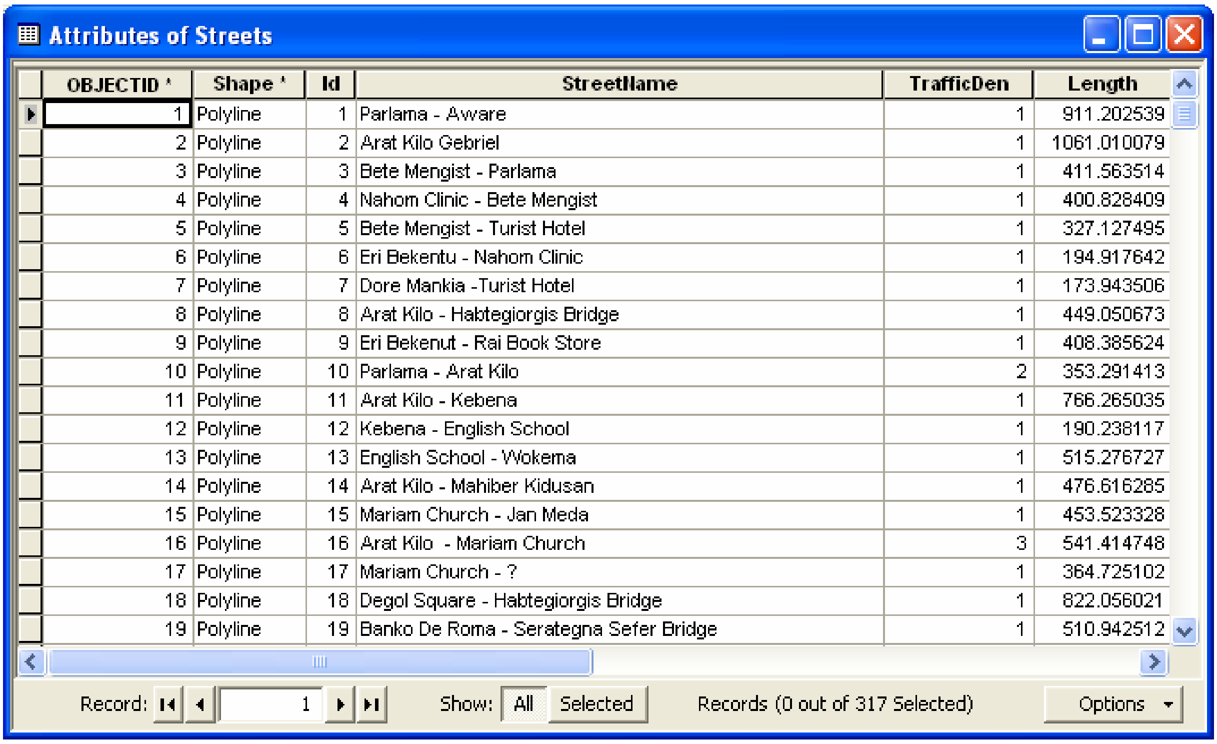


Figure 23: Digitized Streets for the Study Area

---

[i] *Digitization is a process of on screen tracing technique the street network displayed in the image*

Next operation on the streets feature was to determine assign attributes of the street. Street attribute includes its name, whether or not its one way or not, its traffic state, its length, and any restriction associated with it like no pedestrian, maximum weight, etc… These attributes, (the attribute table is shown on Figure 24) are very important during driving route advice analysis. The analysis will exclusively depend on these attribute values to get the optimal solution. In setting the attributes of each street one of the challenges faced was all street doesn't have name. To circumvent the situation, as described in Scope and Limitation section of this document, the project has adopted a naming technique. The technique is to name each street popular ground feature (like business, customary local name, etc…) that marks their beginning and ending points. To increase the readability of the street name each name was enclosed in rectangular bracket.



OBJECTID *	Shape *	Id	StreetName	TrafficDen	Length
1	Polyline	1	Parlama - Aware	1	911.202539
2	Polyline	2	Arat Kilo Gebriel	1	1061.010079
3	Polyline	3	Bete Mengist - Parlama	1	411.563514
4	Polyline	4	Nahom Clinic - Bete Mengist	1	400.828409
5	Polyline	5	Bete Mengist - Turist Hotel	1	327.127495
6	Polyline	6	Eri Bekentu - Nahom Clinic	1	194.917642
7	Polyline	7	Dore Mankia -Turist Hotel	1	173.943506
8	Polyline	8	Arat Kilo - Habtegiorgis Bridge	1	449.050673
9	Polyline	9	Eri Bekenut - Rai Book Store	1	408.385624
10	Polyline	10	Parlama - Arat Kilo	2	353.291413
11	Polyline	11	Arat Kilo - Kebena	1	766.265035
12	Polyline	12	Kebena - English School	1	190.238117
13	Polyline	13	English School - Wokema	1	515.276727
14	Polyline	14	Arat Kilo - Mahiber Kidusan	1	476.616285
15	Polyline	15	Mariam Church - Jan Meda	1	453.523328
16	Polyline	16	Arat Kilo - Mariam Church	3	541.414748
17	Polyline	17	Mariam Church - ?	1	364.725102
18	Polyline	18	Degol Square - Habtegiorgis Bridge	1	822.056021
19	Polyline	19	Banko De Roma - Serategna Sefer Bridge	1	510.942512

Figure 24: Street Attributes

6.2.3 *Network Dataset Building*

The network dataset is a special geodata that is by network analysts to determine network edges, junctures, and associated attributes that are considered during the analysis. To generate the network dataset (Figure 25) the above streets feature is used as an input. The network dataset is actually created by a tool provided in ArcCatalog. The tool, named 'New Network

Dataset' wizard has several process steps that enables content authors to convert their street dataset to network dataset.

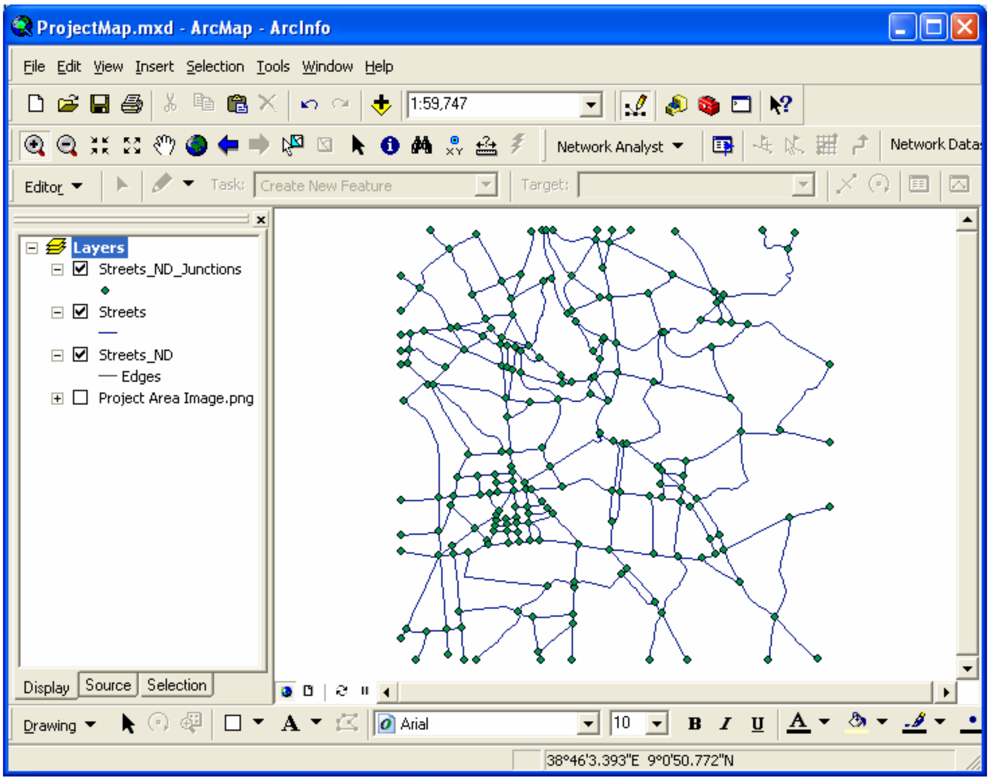


Figure 25: Street Network Data

6.2.4 *Places and Business Location Catalog*

In order to get the best possible shortest path between to points in a given network; the starting and destination points must be given explicitly for the analyst. When seen from the perspective of the network analyst, it is enough for it if it gets latitude and longitude value of both starting and destination points. But this would be too difficult for a user to know his starting and destination lat-lon values.

To solve this problem, the project has used address locator mechanism provided by ArcGIS. In this infrastructure, users will supply location name and it will search the name in an underling points feature geodataset attribute table. Upon identification of a matching name, it will return the location value (lat-lon value) of the corresponding point.

To support the aforementioned locator functionality there was introduced one point feature geodataset. This point feature geodataset is populated with point features representing location of known places, business, buildings, bridges, embassies, customary name of a location, etc… At run time, these points were presented for users to choose form of so that the system will identify form where the user wants to start and to where s/he wants to reach.

6.2.5 *Publishing the Map Service*

After the base map, the streets network and the associated network dataset, and addressing functionality are implemented they were published on ArcGIS server to be consumed by web clients. Figure 26 shows the service being staged on GIS. As illustrated in the ***Location*** drop down control, the server name hosting the mapping and the geocoding (addressing) service is depicted. The status of each service and the number of instances running on the GIS server is shown in the ***content*** tab at the lower right part of the picture.

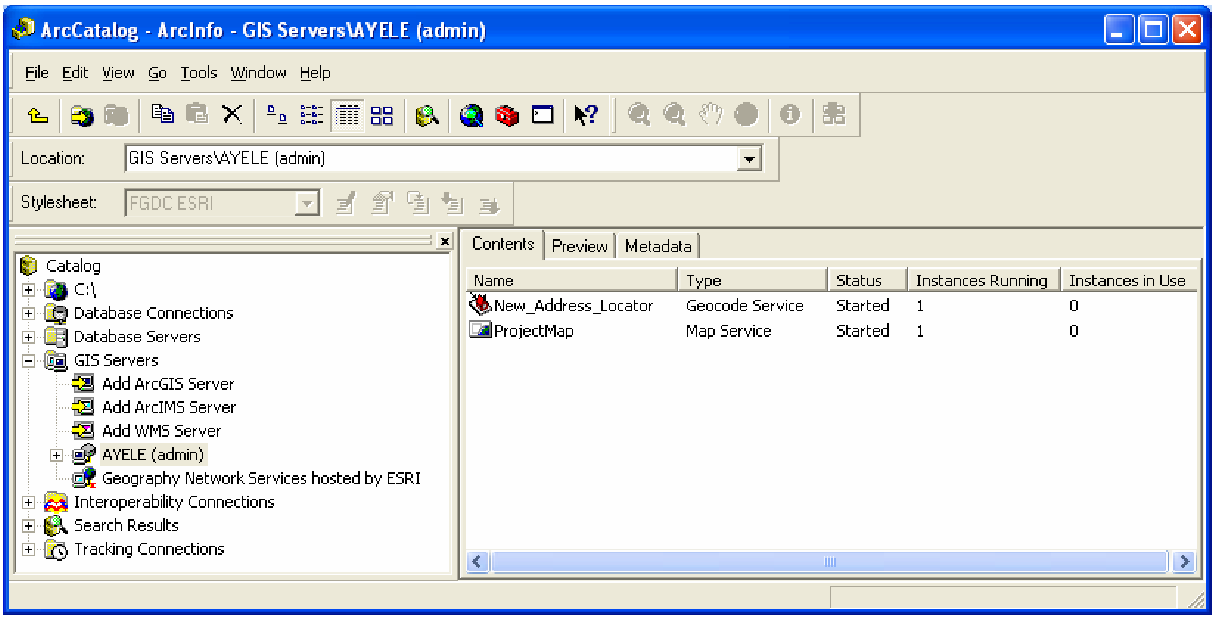


Figure 26: Instance of Map Service running on ArcGIS Server

6.2.6 *Client UI Development*

After the map service is mounted on the ArcGIS server the client user interface and the associated class that tap the map service functionality were implemented in Visual Studio 2005 using ASP.NET 2.0. Here the core of the implementation code writing in the project is found here.

As it mentioned in section 6.1 ESRI, through its ADF for Microsoft Visual Studio, provides a pre-built web mapping controls that are seamlessly integrated with Visual Studio 2005. As shown in figure 27 in the left most panel (toolbox), there are a number of pre-built web controls to facilitate web mapping site development. ArcGIS Web Control groups provide complex web page controls that supports and communicates with ArcGIS server and displays result in web browsers. By inserting this web controls and setting their property and using their method, the projects user interface was realized. The code written for the most important class is annexed

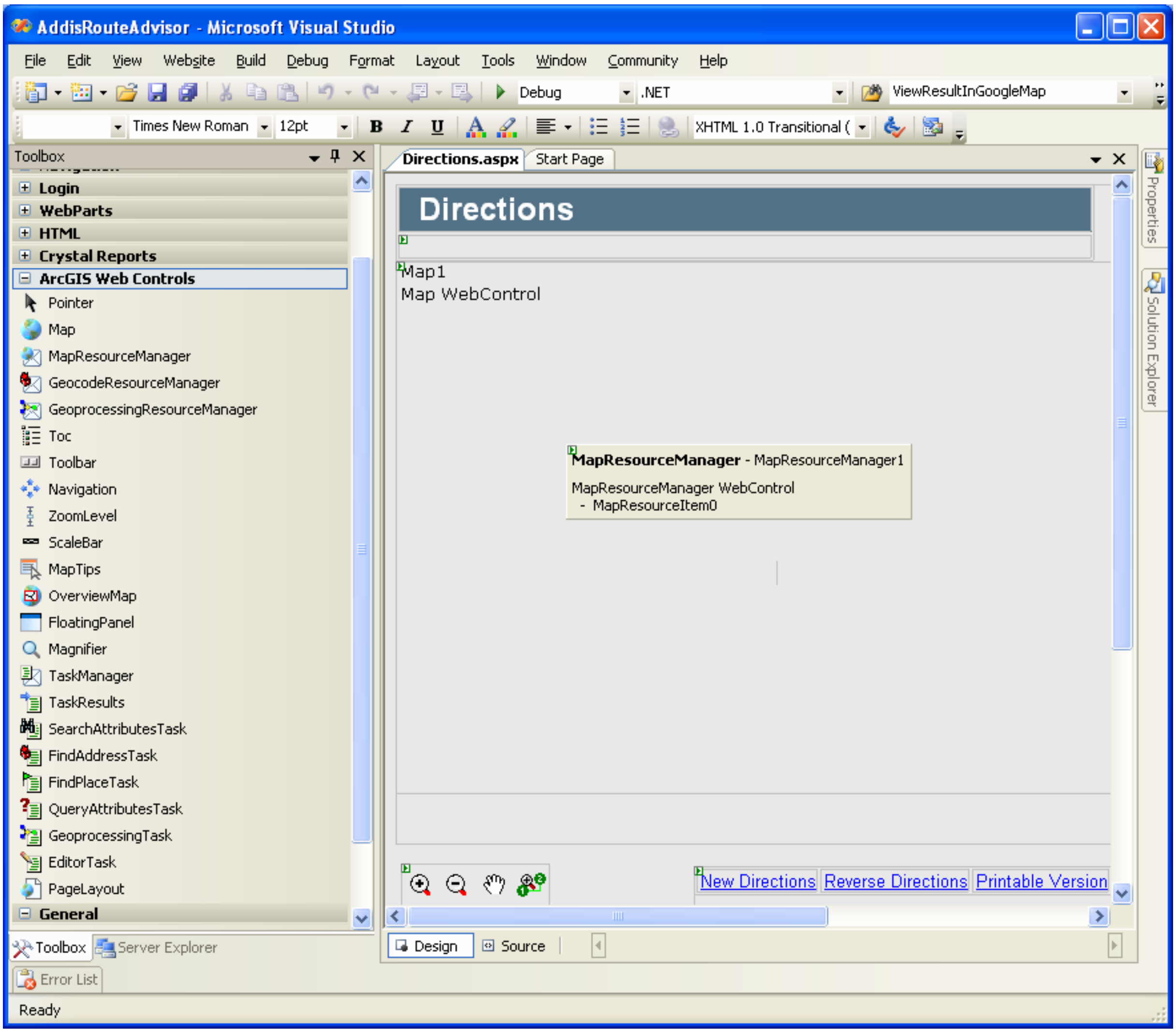


Figure 27: Sample Page and ArcGIS Web Controls used to Create the GUI

Chapter 7

Conclusion and Recommendations

As it applies to any map, web map has very wide spectrum of applicability which includes locating stuff and helping one to figure out how to reach it. Web mapping usage and its integration with other social or commercial sites is continuously growing. Besides, the technology itself is evolving continuously and is getting more and more richer in functionalities. Even though the technology has its own set of requirement unlike the customary web sites, it's appealing visual richness and dynamism is attraction to wide mass of Internet surfers. Seeing this lucrative demand, contemporary Internet business giants are investing a lot to deliver the most comprehensive service for Internet user. This is evidenced by the intensive competition that is going among the mentioned entities.

From the research made in searching feasible means of implementing the project, it can easily be learnt that there are wide array of web mapping technology with lots of accompanying algorithms that can be used to deliver various web mapping functionalities. The capability of these algorithms can be exploited to deliver locally contextualized solutions that enhance the day to day life of residents. As a result; this project has attempted to use at least one of the existing algorithms; which is identifying optimal route between a given starting point and destination point that takes into consideration various costs (like distance) and restrictions (like traffic density), and applied that to Addis Ababa city. The achievement has shade a light on the potential as to what mapping service can deliver useful service for the local people.

Most of the challenges encountered during the project life are extrinsic in nature. These include the lack of standard naming scheme for streets and addressing mechanism for locations with in the city. In Addis Ababa some streets have names but other does not have. This coupled with lack of standardized business location were difficult to structure them and adopt into the project. As a result the project has resorted to implement ad hoc street naming and addressing structure.

The other challenge was with regard to the attempt to inculcate traffic information into to the system. Since there is no realtime traffic information source the project was forced to adopt abstraction mechanism

In light of the aforementioned achievements and constraining challenges; there are lots more to do in the future. First and for most the existing project can be expanded in terms of area coverage to cover the entire city. As mentioned in the Scope and Limitation section the project is restricted to 25 km sq area due to budget and time limitation.

This project can also be extended to include other web mapping related services apart from the route find process by implementing an integrating interface (API) with other online services. For example the site may be integrated with business directory services, historic and tourist attraction sites, advertising services, etc….

The other point to consider as a future work is implementing realtime traffic information monitoring scheme and linking it to the system. The project relied on using an XML based street traffic density status information which will be uploaded as per need by system administrator. This can be automated by implementing a traffic sensor network and embedding the result into the system through web service that delivers the result in XML format.

The system can also be upgraded to support dynamic street naming mechanism. New streets may be constructed and existing may be renovated. In like manner names, even though it is too unlikely, may be changed. Such kinds of dynamisms are not captured in the current version and hence can be considered as a future work to inculcate such functionalities into the system.

The other work that can boost the usability of the system by wide mass of user is implementation of voice direction support. As it has been discussed, the system delivers its output as printable version which explicitly requires reading capability. To assist disabled users and those using the site while driving, voice support is a good option.

Annex

Implementation of NetworkAnalystRouteResult class

```
using System;
using System.Data;
using System.Collections.Specialized;
using ESRI.ArcGIS.ADF.Web.Geometry;


namespace ESRI.ArcGIS.Server.Web.NetworkAnalyst
{
      /// <summary>
      /// Result from a Network Analyst Solve operation on Route layers.
      /// </summary>
      public class NetworkAnalystRouteResult
      {
            /// <summary>
            /// Constructor.
            /// </summary>
            public NetworkAnalystRouteResult()
            {
            }

            #region Member Vars
            StringDictionary m_summary;
            Envelope m_extent;
            DataTable m_directions;
            Envelope[] m_directionsExtents;
            int m_routeID;
            #endregion

            #region Properties

            /// <summary>
            /// Summary information on this route.
            /// </summary>
            public StringDictionary Summary
            {
                  get { return m_summary; }
                  set { m_summary = value; }
            }

            /// <summary>
            /// The extent of this route.
            /// </summary>
            public Envelope RouteExtent
            {
                  get { return m_extent; }
                  set { m_extent = value; }
            }

            /// <summary>
            /// The object id of the route feature
            /// </summary>
            public int RouteID
            {
                  get { return m_routeID; }
                  set { m_routeID = value; }
```

```
            }

            /// <summary>
            /// The step-by-step directions for this route.
            /// </summary>
            public DataTable Directions
            {
                  get { return m_directions; }
                  set { m_directions = value; }
            }

            /// <summary>
            /// Envelope for each step in the directions.  Reference by row
index.
            /// </summary>
            public Envelope[] StepExtents
            {
                  get { return m_directionsExtents; }
                  set { m_directionsExtents = value; }
            }

            #endregion

      }
}
```

Implementation of **CustomTools** Class

```
using System;
using System.Data;
using System.Drawing;
using System.Collections;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;
using ESRI.ArcGIS.ADF.Web;
using ESRI.ArcGIS.ADF.Web.DataSources.ArcGISServer;
using ESRI.ArcGIS.ADF.Web.UI.WebControls.Tools;
using ESRI.ArcGIS.ADF.Web.UI.WebControls;
using ESRI.ArcGIS.ADF.Web.DataSources;
using ESRI.ArcGIS.ADF.Web.Geometry;
using ESRI.ArcGIS.Carto;
using ESRI.ArcGIS.GlobeCore;
using ESRI.ArcGIS.Server;
using ESRI.ArcGIS.Server.Web.NetworkAnalyst;
using ESRI.ArcGIS.NetworkAnalyst;

namespace RouteFinder
{
	public class CustomTools : IMapServerCommandAction,
IMapServerToolAction
	{
		public void ServerAction(ToolbarItemInfo info)
		{
			ESRI.ArcGIS.ADF.Web.UI.WebControls.Map mapctrl =
(ESRI.ArcGIS.ADF.Web.UI.WebControls.Map)info.BuddyControls[0];

			switch (info.Name)
			{
				case "FullRoute":
					Envelope fullextent =
mapctrl.Page.Session["PathExtent"] as Envelope;
					if (fullextent != null)
					{
						mapctrl.Extent = fullextent;
					}
					break;
				case "ZoomToStep":
					HtmlInputHidden Step =
(HtmlInputHidden)mapctrl.Page.FindControl("Step");
					int step = Convert.ToInt32(Step.Value);
					if (step < 1) step = 1;
					Envelope[] extents =
mapctrl.Page.Session["DirectionExtents"] as Envelope[];
					if (extents != null)
					{
						Envelope extent = extents[step];
						mapctrl.Extent = extent;
					}
					break;
				case "ReverseDirections":
					IBaseRouteFinderPage bpi =
(IBaseRouteFinderPage)mapctrl.Page;
```

```
                              // get coords and locations from session
                              double toX =
Convert.ToDouble(mapctrl.Page.Session["FromX"]);
                              double toY =
Convert.ToDouble(mapctrl.Page.Session["FromY"]);
                              double fromX =
Convert.ToDouble(mapctrl.Page.Session["ToX"]);
                              double fromY =
Convert.ToDouble(mapctrl.Page.Session["ToY"]);
                              string fromAddress =
Convert.ToString(mapctrl.Page.Session["ToAddress"]);
                              string toAddress =
Convert.ToString(mapctrl.Page.Session["FromAddress"]);
                              // solve the route and display

                              if (fromX == 0.0 && toX == 0.0 && fromY ==
0.0 && toY == 0.0)
                                    return;

                              NetworkAnalystRouteResult result =
bpi.SolveRoute(fromX, fromY, toX, toY, fromAddress, toAddress);
                              bpi.DisplayDirections(result, fromAddress,
toAddress);
                              mapctrl.Page.Session["FromX"] = fromX;
                              mapctrl.Page.Session["FromY"] = fromY;
                              mapctrl.Page.Session["ToX"] = toX;
                              mapctrl.Page.Session["ToY"] = toY;
                              mapctrl.Page.Session["FromAddress"] =
fromAddress;
                              mapctrl.Page.Session["ToAddress"] =
toAddress;

                              break;
                        case "NewDirections":
                              string url = "Default.aspx?Reset=true";
                              mapctrl.Page.Response.Redirect(url, true);
                              break;
                        case "PrintVersion":
                              MapFunctionality mf =
(MapFunctionality)mapctrl.GetFunctionality(0);
                              mapctrl.ImageFormat =
ESRI.ArcGIS.ADF.Web.WebImageFormat.PNG8;
                              ESRI.ArcGIS.ADF.Web.MapImage mi =
mf.DrawExtent(mapctrl.Extent);

                              MimeData md = mi.MimeData;
                              mapctrl.Page.Session["mymimedata"] = md;


      mapctrl.Page.Response.Write("<script>window.open('" +
"PrintPage.aspx', 'PrintWindow', 'dependent=yes ,width=800, height=500,
status=no, toolbar=no, menubar=yes, location=no, resizable=yes,
scrollbars=yes'); </script>");

                              break;

                case "ViewResultInGoogleMap":

                    break;
                  }
```

```
            }

            #region IMapServerToolAction Members

            public void ServerAction(ToolEventArgs args)
            {
                  throw new System.Exception("The method or operation is
not implemented.");
            }

            #endregion
      }
}
```

Implementation of NetworkAnalystUtility Class

```
using System;
using ESRI.ArcGIS.Server;
using ESRI.ArcGIS.esriSystem;
using ESRI.ArcGIS.Geometry;
using ESRI.ArcGIS.Carto;
using ESRI.ArcGIS.Geodatabase;
using ESRI.ArcGIS.NetworkAnalyst;
using System.Collections;
using System.Data;
using System.Collections.Specialized;
using ESRI.ArcGIS.ADF.Web.DataSources.ArcGISServer;
using ESRI.ArcGIS.ADF.Web.Geometry;

namespace ESRI.ArcGIS.Server.Web.NetworkAnalyst
{
	/// <summary>
	/// Utility class for working with Network analyst
	/// </summary>
	public class NetworkAnalystUtility
	{
		/// <summary>
		/// Constructor
		/// </summary>
		public NetworkAnalystUtility()
		{
		}

		/// <summary>
		/// Add network location
		/// </summary>
		/// <param name="naContext">Layer to add to</param>
		/// <param name="locationClassName">Type of location to
add</param>
		/// <param name="point">Geometry of location</param>
		/// <param name="locationName">Name of location</param>
		/// <param name="tolerance">Tolerance to use</param>
		/// <returns>Whether the location was added</returns>
		public static bool AddLocation(INAContext naContext, string
locationClassName, IPoint point, string locationName, double tolerance)
		{
			// Get NAClass
			INAClass naClass =
naContext.NAClasses.get_ItemByName(locationClassName) as INAClass;
			IFeatureClass featureClass = naClass as IFeatureClass;
			IFields fields = featureClass.Fields;

			#region Find the network location using the locator

			INALocator naLocator = naContext.Locator;
			naLocator.SnapTolerance = tolerance;
			INALocation naLocation = null;
			IPoint outPoint = null;
			double distanceFromPoint = 0;
			naLocator.QueryLocationByPoint( point, ref naLocation,
ref outPoint, ref distanceFromPoint);

			#endregion
```

```
                    // Get total count of features in class (needed for
Sequence)
                    int count = featureClass.FeatureCount(null);

                    // Create feature and set shape
                    IFeature feature = featureClass.CreateFeature();
                    ((IRowSubtypes)feature).InitDefaultValues();
                    feature.Shape = point;

                    #region Set field values for NALocation, Name, Sequence,
Status

                    // Set NALocation
                    ((INALocationObject)feature).NALocation = naLocation;

                    // Set name field
                    if (locationName.Trim().Length > 0)
                        feature.set_Value(fields.FindField("Name"),
locationName);

                    // If Stops, set Sequence field (required to go from 1 to
N)
                    int sequenceFieldIndex = fields.FindField("Sequence");
                    if (sequenceFieldIndex >= 0)
                        feature.set_Value(sequenceFieldIndex, count + 1);

                    // Set status if not located
                    if (naLocation.IsLocated == false)
                        feature.set_Value(fields.FindField("Status"),
esriNAObjectStatus.esriNAObjectStatusNotLocated);
                    #endregion

                    // Make sure to store the feature
                    feature.Store();

                    return naLocation.IsLocated;
            }

            /// <summary>
            /// Get directions
            /// </summary>
            /// <param name="naContext">The network analyst layer</param>
            /// <param name="locationClassName">The output class name, e.g.
Routes</param>
            /// <param name="serverContext">The server context</param>
            /// <returns>Route results</returns>
            public static NetworkAnalystRouteResult
GetDirections(INAContext naContext, string locationClassName,
IServerContext serverContext)
            {
                    #region Check if we have results
                    INATraversalResultQuery result = naContext.Result as
INATraversalResultQuery;
                    if (result == null) return null;
                    int resultCount =
result.get_FeatureClass(esriNetworkElementType.esriNETEdge).FeatureCount(nu
ll);
                    if (resultCount == 0) return null;
                    #endregion
```

```
                  INAClass naClass =
naContext.NAClasses.get_ItemByName(locationClassName) as INAClass;

                  #region Get Result Features
                  IFeatureClass fClass = naClass as IFeatureClass;
                  ISet resultSet;
                  IQueryFilter filter;
                  if (serverContext != null)
                  {
                        resultSet =
serverContext.CreateObject("esriSystem.Set") as ISet;
                        filter =
serverContext.CreateObject("esriGeodatabase.QueryFilter") as IQueryFilter;
                  }
                  else
                  {
                        resultSet = new SetClass() as ISet;
                        filter = new QueryFilterClass() as IQueryFilter;
                  }

                  ICursor cursor = fClass.Search(null, false) as ICursor;
                  IRow row = cursor.NextRow();
                  while (row != null)
                  {
                        resultSet.Add(row);
                        row = cursor.NextRow();
                  }
                  #endregion

                  #region Prepare Directions
                  INamedSet namedSet = naContext.Agents;
                  string agentName = namedSet.get_Name(0);
                  INAStreetDirectionsAgent agent =
namedSet.get_ItemByName(agentName) as INAStreetDirectionsAgent;
                  ITrackCancel track;
                  if (serverContext != null)
                        track =
serverContext.CreateObject("esriDisplay.CancelTracker") as ITrackCancel;
                  else
                        track = new
ESRI.ArcGIS.Display.CancelTrackerClass() as ITrackCancel;
                  INAAgent naAgent = agent as INAAgent;
                  naAgent.OnResultUpdated();
                  agent.Execute(resultSet, track);
                  #endregion

                  #region Get Directions

                  ArrayList nasResultItems = new ArrayList();
                  INAStreetDirectionsContainer container =
agent.DirectionsContainer;
                  int count = 0;
                  string columnName, columnValue;
                  DataRow dataRow;
                  IEnvelope envelope;

                  for (int i = 0; i < container.DirectionsCount; i++)
                  {
                        NetworkAnalystRouteResult resultItem = new
NetworkAnalystRouteResult();
```

```
                        INAStreetDirections streetDirections =
container.get_Directions(i);
                        INAStreetDirection streetDirection =
streetDirections.Summary;
                        esriDirectionsStringType type =
esriDirectionsStringType.esriDSTGeneral;
                        #region Get Summary
                        StringDictionary summary = new StringDictionary();
                        summary["Route"] = streetDirections.RouteName;
                        count = streetDirection.StringCount;
                        for (int k = 0; k < count; k++)
                        {
                              type = streetDirection.get_StringType(k);
                              columnName = GetDirectionsStringType(type);
                              columnValue = summary[columnName] as string;
                              if (columnValue != null && columnValue.Length
> 0)
                                    summary[columnName] = columnValue +
"<br>" + streetDirection.get_String(k);
                              else
                                    summary[columnName] =
streetDirection.get_String(k);
                        }
                        envelope = streetDirection.Envelope;

                        // Make sure we create a
DefinitionSpatialReferenceInfo object in the case
                        // where the spatial reference has a factory code.
Use it for the envelope's
                        // CoordinateSystem.  Otherwise, the envelope's
CoordinateSystem will be an
                        // IDSpatialReferenceInfo object which may cause
the map to not draw correctly.
                        ISpatialReference spatialReference =
envelope.SpatialReference;

      ESRI.ArcGIS.ADF.Web.SpatialReference.SpatialReferenceInfo
spatialReferenceInfo = null;
                        if (spatialReference != null)
                        {
                              if (spatialReference.FactoryCode != 0)
                              {
                                    IESRISpatialReferenceGEN esriSR =
(IESRISpatialReferenceGEN)envelope.SpatialReference;
                                    string definition;
                                    int bWrote;
                                    esriSR.ExportToESRISpatialReference(out
definition, out bWrote);
                                    spatialReferenceInfo = new
ESRI.ArcGIS.ADF.Web.SpatialReference.DefinitionSpatialReferenceInfo(definit
ion);
                              }
                        }

                        envelope.Expand(1.25, 1.25, true);
                        ESRI.ArcGIS.ADF.Web.Geometry.Envelope adf_envelope
= Converter.FromIEnvelope(envelope);
                        if (spatialReferenceInfo != null)

      adf_envelope.SpatialReference.CoordinateSystem =
spatialReferenceInfo;
```

```
                        resultItem.RouteExtent = adf_envelope;
                        resultItem.RouteID = streetDirections.RouteID;
                        #endregion

                        #region Get Directions
                        DataTable directions = new DataTable("Directions");
                        directions.Columns.Add("Step", typeof(string));
                        directions.Columns.Add("Directions",
typeof(string));
                        directions.Columns.Add("Length", typeof(string));
                        directions.Columns.Add("Summary", typeof(string));
                        directions.Columns.Add("Time", typeof(string));
                        directions.Columns.Add("Type", typeof(string));
                        directions.Columns.Add("Cumulative Length",
typeof(string));
                        //directions.Columns.Add("ETA", typeof(string));
                        //directions.Columns.Add("Service Tim",
typeof(string));
                        //directions.Columns.Add("Time Window",
typeof(string));
                        //directions.Columns.Add("Violation Time",
typeof(string));
                        //directions.Columns.Add("Wait Time",
typeof(string));

                        resultItem.StepExtents = new
ESRI.ArcGIS.ADF.Web.Geometry.Envelope[streetDirections.DirectionCount];

                        for (int j = 0; j <
streetDirections.DirectionCount; j++)
                        {
                              streetDirection =
streetDirections.get_Direction(j);
                              count = streetDirection.StringCount;
                              dataRow = directions.NewRow();
                              directions.Rows.Add(dataRow);

                              for (int k = 0; k < count; k++)
                              {
                                    #region Get type of direction; column
name
                                    type =
streetDirection.get_StringType(k);
                                    columnName =
GetDirectionsStringType(type);
                                    if (type ==
esriDirectionsStringType.esriDSTArrive ||
                                          type ==
esriDirectionsStringType.esriDSTDepart)
                                    {
                                          directions.Rows[j]["Type"] =
columnName;
                                          columnName = "Directions";
                                    }
                                    else if (type ==
esriDirectionsStringType.esriDSTGeneral)
                                    {
                                          directions.Rows[j]["Type"] =
"General";
                                    }
                                    #endregion
```

```
                                    #region Populate column
                                    directions.Rows[j]["Step"] = j + 1;
                                    columnValue =
directions.Rows[j][columnName] as string;
                                    if (columnValue != null &&
columnValue.Length > 0)
                                          directions.Rows[j][columnName] =
columnValue + "<br>" + streetDirection.get_String(k);
                                    else
                                          directions.Rows[j][columnName] =
streetDirection.get_String(k);
                                    #endregion
                              }
                              envelope = streetDirection.Envelope;
                              envelope.Expand(1.25, 1.25, true);
                              adf_envelope =
Converter.FromIEnvelope(envelope);
                              if (spatialReferenceInfo != null)

      adf_envelope.SpatialReference.CoordinateSystem =
spatialReferenceInfo;
                              resultItem.StepExtents[j] = adf_envelope;
                        }
                        #endregion

                        resultItem.Directions = directions;
                        resultItem.Summary = summary;
                        nasResultItems.Add(resultItem);
                  }
                  #endregion

                  // Assume only 1 route in the analysis problem so just
return the directions for array[0]
                  return nasResultItems[0] as NetworkAnalystRouteResult;
            }


            /// <summary>
            /// Gets the column name for directions string type
            /// </summary>
            /// <param name="type">The type to get string for.</param>
            /// <returns>The string for direction type.</returns>
            public static string
GetDirectionsStringType(esriDirectionsStringType type)
            {
                  switch (type)
                  {
                        case esriDirectionsStringType.esriDSTArrive:
                              return "Arrive";
                        case esriDirectionsStringType.esriDSTDepart:
                              return "Depart";
                        case esriDirectionsStringType.esriDSTGeneral:
                              return "Directions";
                        case esriDirectionsStringType.esriDSTLength:
                              return "Length";
                        case esriDirectionsStringType.esriDSTSummary:
                              return "Summary";
                        case esriDirectionsStringType.esriDSTTime:
                              return "Time";
```

```
                        case 
esriDirectionsStringType.esriDSTCumulativeLength:
                              return "Cumulative Length";
                        case 
esriDirectionsStringType.esriDSTEstimatedArrivalTime:
                              return "ETA";
                        case esriDirectionsStringType.esriDSTServiceTime:
                              return "Service Time";
                        case esriDirectionsStringType.esriDSTTimeWindow:
                              return "Time Window";
                        case esriDirectionsStringType.esriDSTViolationTime:
                              return "Violation Time";
                        case esriDirectionsStringType.esriDSTWaitTime:
                              return "Wait Time";
                        default:
                              return "Default";
                  }
            }

            /// <summary>
            /// Gets the layer using the layer id
            /// </summary>
            /// <param name="map">The map to which the layer 
belongs</param>
            /// <param name="layerID">The id of the layer</param>
            /// <returns>The layer at the id.</returns>
            public static ILayer LayerFromLayerID(IMap map, int layerID)
            {
                  UID uidINALayer = new UIDClass();
                  uidINALayer.Value = "{667B776B-5905-4450-9C94-
18B214ECE8FB}";

                  IEnumLayer elayers = map.get_Layers(uidINALayer, true);
                  ILayer elayer = elayers.Next();
                  int i = 0;
                  while (elayer != null)
                  {
                        if (i == layerID)
                              return elayer;
                        ++i;
                        elayer = elayers.Next();
                  }
                  return null;
            }
      }
}
```

References

1 Mitchel T, ***Web Mapping Illustrated***, O'Reilly Media, Inc , 2005

2 ***http://www.maphistory.info***, Last visited on 20, May 2008

3 Erle S., Gibson R , ***Google Map Hacks***, O'Reilly Media, Inc. 2006

4 ***http://www.google.com/corporate/tenthings.html***, Last visited on 9 March 2008

5 ***http://maps.google.com***, Last visited on 9 March 2008

6 ***http://reviews.cnet.com/online-software-services/google-maps/4505-9239_7-31591128.html***, Last visited on 8 March 2008

7 ***http://www.techcrunch.com/2005/11/02/new-yahoo-maps-shows-power-of-flash/***, Last visited on 5 March 2008

8 ***http://reviews.cnet.com/online-software-services/windows-live-local-beta/4505-9239_7-31631858.html***, Last visited on 8 March 2008

9 ***http://reviews.cnet.com/online-software-services/ask-maps-and-directions/4505-9239_7-31902760.html***, Last visited on 8 March 2008

10 ***http://reviews.cnet.com/sonline-software-services/mapquest/4505-9239_7-31880451.html***, Last visited on 9 March 2008

11 Kraak M., Allan B, ***Web Cartography; Development and Prospects***, 2000

12 ***http://www.web-mapping.xfrolis.info***, Last visited on 20 May 2008

13 ***http://earth.images.alaska.edu/geog493/presentations/11_webMaps.ppt*** , Last visited 20 May 2008

DECLARATION

I, the undersigned, declare that this project is my original work and has not been presented for a dgree in any other university, and that all source of materials used for the projecte have been duly acknowledged.

Declared by:

Name: ______________________________________

Signature ______________________________________

Date ______________________________________

Confirmed by advisor:

Name: ______________________________________

Signature ______________________________________

Date ______________________________________

Place and date of submission: Addis Ababa, July 2008.